3

CFM03348
US
Appln. No. 10/789,007
GAU: NYA.

日本国特許庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application: ２００３年　７月２４日

出願番号
Application Number: 特願２００３－２０１１６２

[ST. 10/C]: [ＪＰ２００３－２０１１６２]

出願人
Applicant(s): キヤノン株式会社

２００４年　１月　６日

特許庁長官
Commissioner,
Japan Patent Office 今井康夫

出証番号　出証特２００３－３１０８７７１

【書類名】　特許願
【整理番号】　255622
【提出日】　平成15年 7月24日
【あて先】　特許庁長官殿
【国際特許分類】　G06F　3/00
【発明の名称】　符号データ作成方法及び装置
【請求項の数】　13
【発明者】
【住所又は居所】　東京都大田区下丸子３丁目３０番２号　キヤノン株式会社内
【氏名】　榎田　幸
【特許出願人】
【識別番号】　000001007
【氏名又は名称】　キヤノン株式会社
【代理人】
【識別番号】　100076428
【弁理士】
【氏名又は名称】　大塚　康徳
【電話番号】　03-5276-3241
【選任した代理人】
【識別番号】　100112508
【弁理士】
【氏名又は名称】　高柳　司郎
【電話番号】　03-5276-3241
【選任した代理人】
【識別番号】　100115071
【弁理士】
【氏名又は名称】　大塚　康弘
【電話番号】　03-5276-3241

【選任した代理人】

【識別番号】　100116894

【弁理士】

【氏名又は名称】　木村　秀二

【電話番号】　03-5276-3241

【先の出願に基づく優先権主張】

【出願番号】　特願2002-356738

【出願日】　平成14年12月 9日

【手数料の表示】

【予納台帳番号】　003458

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【包括委任状番号】　0102485

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　符号データ作成方法及び装置

【特許請求の範囲】

【請求項１】　サーバが管理する符号データのうち、断片的な第１の符号データを格納した格納手段を備えるクライアントにおいて、ＪＰＥＧ２０００符号データを作成する符号データ作成方法であって、

前記クライアントにおいて前記ＪＰＥＧ２０００符号データの作成に必要となる符号データと、前記格納手段に格納された前記第１の符号データとから、不足する第２の符号データを算出する算出工程と、

前記サーバに対して、算出された前記第２の符号データを要求する要求工程と、

前記サーバから、前記第２の符号データを取得する取得工程と、

取得された前記第２の符号データを前記格納手段に格納する格納工程と、

取得された前記第２の符号データのヘッダ情報を解析して、前記符号データを複数の独立した符号データに分割する分割工程と、

前記分割工程で分割された単位毎に、前記独立した符号データを構成する全ての符号データが前記格納手段に格納されているか否かを判定する判定工程と、

前記独立した符号データを構成する全ての符号データが格納されていない場合、格納されていない部分にダミー符号データを格納するダミー格納工程と、

前記格納手段に格納された符号データをＪＰＥＧ２０００符号データとして出力する出力工程と

を有することを特徴とする符号データ作成方法。

【請求項２】　前記判定工程によって、前記独立した符号データを構成する全ての符号データが前記格納手段に格納されていると判定された場合、該格納手段に格納された該符号データを前記分割工程で分割された前記独立した符号データに置き換える置換工程をさらに有することを特徴とする請求項１に記載の符号データ作成方法。

【請求項３】　前記符号データが、パケット単位で取り扱われることを特徴とする請求項１又は２に記載の符号データ作成方法。

【請求項４】　前記ダミー符号データが、ＪＰＥＧ２０００で規定されているzero length packetデータであることを特徴とする請求項１から３までのいずれか１項に記載の符号データ作成方法。

【請求項５】　前記分割工程が、所定サイズのタイル単位で前記符号データを分割することを特徴とする請求項１から４までのいずれか１項に記載の符号データ作成方法。

【請求項６】　前記独立した符号データのそれぞれのヘッダ情報を前記分割工程で分割された前記タイル単位のサイズに変更する変更工程をさらに有することを特徴とする請求項１から５までのいずれか１項に記載の符号データ作成方法。

【請求項７】　前記クライアントが画像データを表示する表示手段をさらに備え、前記第１の符号データが前記画像データの符号データであって、

前記表示手段に表示された画像データの表示領域の移動又は拡大表示に応じて、前記算出工程における前記必要となる符号データを設定する設定工程と、

前記出力工程で出力された前記ＪＰＥＧ２０００符号データをデコードするデコード工程と、

デコードされた画像データを前記表示手段に画面表示する表示工程と

をさらに有することを特徴とする請求項１から６までのいずれか１項に記載の符号データ作成方法。

【請求項８】　前記判定工程によって、前記独立した符号データを構成する全ての符号データが前記格納手段に格納されていると判定された場合、前記分割工程で分割された単位ごとに、該単位に関するメインヘッダをそのまま出力し、該単位のインデックスを所定のインデックスに置換し、該単位に含まれる符号化データ本体を前記分割工程で分割された前記独立した符号データに置き換える置換手段をさらに備えることを特徴とする請求項１に記載の符号データ作成方法。

【請求項９】　前記分割工程で分割された少なくとも２以上の単位について、前記判定工程、前記ダミー格納工程、前記置換工程及び前記出力工程を並列処理することを特徴とする請求項２に記載の符号データ作成方法。

【請求項１０】　第１のコンピュータが管理する符号データのうち、断片的

な第１の符号データを格納した格納手段を備える第２のコンピュータにおける符号データ作成装置であって、

前記第１のコンピュータに管理された前記符号データのうち、断片的な第１の符号データを格納する第１の格納手段と、

前記第２のコンピュータにおいてＪＰＥＧ２０００符号データの作成に必要となる符号データと、前記格納手段に格納された前記第１の符号データとから、不足する第２の符号データを算出する算出手段と、

前記第１のコンピュータに対して、算出された前記第２の符号データを要求する要求手段と、

前記第１のコンピュータから、前記第２の符号データを取得する取得手段と、

取得された前記第２の符号データを格納する第２の格納手段と、

取得された前記第２の符号データのヘッダ情報を解析して、前記符号データを複数の独立した符号データに分割する分割手段と、

前記分割手段で分割された単位毎に、前記独立した符号データを構成する全ての符号データが前記第１又は第２の格納手段に格納されているか否かを判定する判定手段と、

前記独立した符号データを構成する全ての符号データが格納されていない場合、格納されていない部分にダミー符号データを格納する第３の格納手段と、

前記第１、第２及び第３の格納手段に格納された符号データを用いてＪＰＥＧ２０００符号データを作成する作成手段と

を備えることを特徴とする符号データ作成装置。

【請求項１１】　前記第１及び第２のコンピュータが、ネットワークを介して互いに通信可能であることを特徴とする請求項９記載の符号データ作成装置。

【請求項１２】　第１のコンピュータが管理する符号データのうち、断片的な第１の符号データを格納した格納手段を備える第２のコンピュータに、ＪＰＥＧ２０００符号データを作成させるためのプログラムであって、

前記第２のコンピュータにおいて前記ＪＰＥＧ２０００符号データの作成に必要となる符号データと、前記格納手段に格納された前記第１の符号データとから、不足する第２の符号データを算出する算出手順と、

前記第１のコンピュータに対して、算出された前記第２の符号データを要求する要求手順と、

前記第１のコンピュータから、前記第２の符号データを取得する取得手順と、

取得された前記第２の符号データを前記格納手段に格納する格納手順と、

取得された前記第２の符号データのヘッダ情報を解析して、前記符号データを複数の独立した符号データに分割する分割手順と、

前記分割手順で分割された単位毎に、前記独立した符号データを構成する全ての符号データが前記格納手段に格納されているか否かを判定する判定手順と、

前記独立した符号データを構成する全ての符号データが格納されていない場合、格納されていない部分にダミー符号データを格納するダミー格納手順と、

前記格納手段に格納された符号データをＪＰＥＧ２０００符号データとして出力する出力手順と

を実行させるためのプログラム。

【請求項１３】　請求項１２記載のプログラムを格納したことを特徴とするコンピュータ読み取り可能な記録媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、ネットワークを介して受信した断片的な画像データから符号データを作成する技術に関する。

【０００２】

【従来の技術】

インターネット上では、WWWサーバにＷｅｂ（ウエブ）ブラウザからアクセスして文書データや画像データ等の情報を閲覧することが盛んに行われている。このように、インターネット上に情報を公開するWWWサーバと、その情報を閲覧するためのクライアントとを含むシステム環境において、各クライアントではウエブブラウザを使用して、WWWサーバによって公開された情報を閲覧することができる。

【０００３】

WWWサーバには、通常、ホームページといわれる公開するための情報をＨＴＭＬで記述した文書が保存されており、それにクライアント側のウエブブラウザがアクセスしてクライアント側のコンピュータに表示する。また、クライアント側のウエブブラウザは、表示しているページ内のリンクを辿っていくことにより、必要な情報を得ることができる。さらに、WWWサーバが管理しているファイルをダウンロードする方法として、File Transfer Protocol （以下、「ＦＴＰ」と略す。）という方法がある。このＦＴＰとは、ネットワークを通して、WWWサーバ上にあるファイルの内容を一度にクライアント側のコンピュータに転送する仕組みのことである。

【０００４】

また、サーバが管理する画像ファイルへ断片的にアクセスして、クライアントで表示するためのプロトコルとして、Flashpix／ＩＩＰがある。このインターネット・イメージング・プロトコル（ＩＩＰ）は、Flashpixという画像データファイルフォーマットに最適なプロトコルになっており、ネットワークを通して画像データの一部分を要求できるものである。このときのアクセスは、Flashpixのタイル単位に行われる。

【０００５】

一方、このFlashpix／ＩＩＰをそのままＪＰＥＧ２０００に適用した場合を考える。ここで、ＪＰＥＧ２０００では、各スケーラビリティ（Scalability）の符号データは、そのスケーラビリティより１つ下のスケーラビリティのデータとの差分データで構成されている。そこで、クライアント側で受信した断片的な符号データをキャッシュしておき、全符号データをデコーダに引き渡して最初から再デコードする方法と、デコーダを途中で止めておいて今回受信した符号データをデコーダに渡し、前回の続きからデコードする方法とがある。

【０００６】

このような方法のうち、マルチ解像度データを有する画像符号データの中から必要な部分のデータのみを取り出して別の符号データに変換する方法が従来から提案されている（例えば、特許文献１参照）。

【０００７】

上記特許文献１においてソースとなる画像データは、ウェーブレット（Wavelet）変換、或いはWavelet-likeな変換を用いて、マルチ解像度のデータを管理することが可能な符号データである。そして、このソースとなる符号データの中から、ユーザが選択した空間的な領域のデータを処理するために必要な符号データを取り出し、１つの独立した符号データに変換する方法が記載されている。尚、この時ソースの符号データから取り出される部分的な符号データは、ＪＰＥＧ２０００のコードブロックに対応しており、今回のユーザからの要求を処理するために必要な符号データを含んでいる。

【０００８】

ここで、サーバから送られてくる符号データをクライアント側で最初から再デコードする場合、それらの断片的な符号データをキャッシュのような仕組みを持たないで、直接、ＪＰＥＧ２０００準拠の１つの符号データファイルに作り直すことが可能である。

【０００９】

【特許文献１】

米国特許第６０４１１４３号明細書

【００１０】

【発明が解決しようとする課題】

しかしながら、サーバ側から送られてくる断片的な符号データは、クライアント側から要求する順に送られてくるため、クライアント側で受信した符号データをＪＰＥＧ２０００準拠の１つの符号データに変換するためには、クライアント側での煩雑な処理が必要であるという問題点がある。

【００１１】

また、サーバから送られてきた断片的な符号データを独自形式のキャッシュファイルとして構成し、クライアント側のＪＰＥＧ２０００デコーダが直接このキャッシュファイルをリードすることも可能である。しかし、この場合、ＪＰＥＧ２０００デコーダには独自のキャッシュファイルをリードするための処理が必要となり、処理が複雑になると共に、汎用のＪＰＥＧ２０００デコーダが使えないという問題点がある。

【００１２】

そこで、ＪＰＥＧ２０００符号データをパケット（packet）単位で受信し、それらのpacketデータをキャッシュしておいて、デコーダへ符号データを渡す際に、まだ受信していないpacketデータ部分に、zero length packet（以下、「ＺＬＰ」と略す。）データを挿入し、既に受信したpacketデータとあわせて、１つのＪＰＥＧ２０００準拠の符号ファイルを作成することで、メインヘッダの書き換えなどの煩雑な処理を行わずに、一般的なＪＰＥＧ２０００デコーダで処理可能なＪＰＥＧ２０００ファイルを作成することが考えられる。

【００１３】

しかし、受信した全てのpacketデータを利用して作成された符号データは、クライアント側のアプリケーションを使っているユーザの要求した画像サイズ、画質及び画像領域等において、ユーザの望む表示形態を考慮していない。そのため、このような符号データを受け取ったデコーダでは、その一部のデータから生成されたデコード結果を利用して表示画面が作成されることがあり、キャッシュされている全ての符号データを使って１つのＪＰＥＧ２０００符号データファイルを作ることは、デコーダ自体の処理にも無駄な処理が生じるという問題がある。

【００１４】

さらに、キャッシュされている符号データが多くなるほど、デコーダへ渡す符号データを作成するときに発生するデータのコピー作業が多くなる。これは、クライアントの表示に要するまでの時間が長くなることにつながり、パフォーマンス低下という問題が生じる。特に、画像の拡大やスクロールという命令をユーザが行うことで、表示に直接必要のないキャッシュデータが多くなり、上記の問題が頻繁に起こることになる。

【００１５】

さらにまた、上記特許文献１には、以下のような問題点がある。すなわち、上記特許文献１に記載の技術をネットワークを通した通信であるＩＩＰにそのまま適用した場合、クライアントからの要求毎に全階層の全コードブロックの符号データを送ることになり、クライアント側で既に受信済みの符号データまでも送信することになる。また、送信する符号データのヘッダ部分は、今回要求した画像

データの情報（例えば、画像サイズ、階層情報等）に書き換えられてしまい、サーバで管理されている符号データの本来の情報を取得することができない。

【００１６】

本発明は、このような事情を考慮してなされたものであり、クライアントにキャッシュされている断片化された符号データと、サーバから必要な分だけ受信した断片化された符号データとを用いて、クライアントにおいて汎用ＪＰＥＧ２０００デコーダで使用可能な符号データであって、当該符号データのデコード及び画像データの表示処理を高速に行うことができる符号データを好適に作成することができる符号データ作成方法及び装置を提供することを目的とする。

【００１７】

【課題を解決するための手段】

上記課題を解決するために、本発明は、サーバが管理する符号データのうち、断片的な第１の符号データを格納した格納手段を備えるクライアントにおいて、ＪＰＥＧ２０００符号データを作成する符号データ作成方法であって、

前記クライアントにおいて必要となる符号データと、前記格納手段に格納された前記第１の符号データとから、不足する第２の符号データを算出する算出工程と、

前記サーバに対して、算出された前記第２の符号データを要求する要求工程と、

前記サーバから、前記第２の符号データを取得する取得工程と、

取得された前記第２の符号データを前記格納手段に格納する格納工程と、

取得された前記第２の符号データのヘッダ情報を解析して、前記符号データを複数の独立した符号データに分割する分割工程と、

前記分割工程で分割された単位毎に、前記独立した符号データを構成する全ての符号データが前記格納手段に格納されているか否かを判定する判定工程と、

前記独立した符号データを構成する全ての符号データが格納されていない場合、格納されていない部分にダミー符号データを格納するダミー格納工程と、

前記格納手段に格納された符号データをＪＰＥＧ２０００符号データとして出力する出力工程と

を有することを特徴とする。

【００１８】

また、本発明に係る上記符号データ作成方法は、前記判定工程によって、前記独立した符号データを構成する全ての符号データが前記格納手段に格納されていると判定された場合、該格納手段に格納された該符号データを前記分割工程で分割された前記独立した符号データに置き換える置換工程をさらに有することを特徴とする。

【００１９】

さらに、本発明は、第１のコンピュータが管理する符号データのうち、断片的な第１の符号データを格納した格納手段を備える第２のコンピュータにおける符号データ作成装置であって、

前記第１のコンピュータに管理された前記符号データのうち、断片的な第１の符号データを格納する第１の格納手段と、

前記第２のコンピュータにおいてＪＰＥＧ２０００符号データの作成に必要となる符号データと、前記格納手段に格納された前記第１の符号データとから、不足する第２の符号データを算出する算出手段と、

前記第１のコンピュータに対して、算出された前記第２の符号データを要求する要求手段と、

前記第１のコンピュータから、前記第２の符号データを取得する取得手段と、

取得された前記第２の符号データを格納する第２の格納手段と、

取得された前記第２の符号データのヘッダ情報を解析して、前記符号データを複数の独立した符号データに分割する分割手段と、

前記分割手段で分割された単位毎に、前記独立した符号データを構成する全ての符号データが前記第１又は第２の格納手段に格納されているか否かを判定する判定手段と、

前記独立した符号データを構成する全ての符号データが格納されていない場合、格納されていない部分にダミー符号データを格納する第３の格納手段と、

前記第１、第２及び第３の格納手段に格納された符号データを用いてＪＰＥＧ２０００符号データを作成する作成手段と

を備えることを特徴とする。

【００２０】

【発明の実施の形態】

以下、図面を参照して、本発明の実施形態について詳細に説明する。

【００２１】

＜第１の実施形態＞

図１は、ネットワーク上に複数のコンピュータが接続されている様子を示す図であり、本発明の第１の実施形態に係るサーバ及びクライアントを含むネットワークシステムの構成を示す概要図である。

【００２２】

図１において、１００はインターネットに代表されるネットワークである。１０１、１０３は、ネットワーク１００に接続されているサーバ・コンピュータであり、画像データを送信するためのＪＰＥＧ２０００用のＩＩＰサーバを始めとして、WWWサーバ機能に必要なソフトウェアが実行されている。また、大量の画像データも格納されている。また、１０２ａ、１０２ｂはクライアント･コンピュータであり、ウエブブラウザやＪＰＥＧ２０００デコーダ等のクライアント側で必要なソフトウェアが実行されている。

【００２３】

図２は、図１のサーバ・コンピュータ１０１、１０３又はクライアント・コンピュータ１０２ａ、１０２ｂの各コンピュータシステムのハードウェア構成の一例を示す図である。図２において、２０１はＣＰＵであり、システム全体の制御等を行っている。２０２はキーボードであり、マウス２０１ａと共に本システムに対して情報等を入力するために使用される。また、２０３は表示部であり、ＣＲＴや液晶ディスプレイ等で構成されている。

【００２４】

一方、２０４はＲＯＭ、２０５はＲＡＭであり、本システムにおける記憶装置を構成しており、システムが実行するプログラムやシステムが利用するデータ等を記憶する。また、２０６はハードディスク装置、２０７はフレキシブルディスクであり、本システムのファイルシステムに使用される外部記憶装置を構成して

いる。さらに、２０８はプリンタである。さらにまた、２０９はネットワークを制御する部分であり、ここからネットワーク１００（例えば、インターネット等）上に接続されているサーバ・コンピュータ等のネットワーク上の資源にアクセスする。

【００２５】

本実施形態では、既に生成済みのＪＰＥＧ２０００ファイルがサーバ・コンピュータで管理されている。一方、クライアント・コンピュータでは、画像表示アプリケーション等を用いてユーザが要求したＪＰＥＧ２０００ファイル内の必要な部分のみを、ＪＰＥＧ２０００用ＩＩＰを使用して上記サーバ・コンピュータにアクセスし、符号データを断片的に受信し、受信した断片的な符号データをクライアント・コンピュータ上の例えばハードディスク２０６等にキャッシュする。

【００２６】

以下では、ハードディスク２０６等にキャッシュされた断片的なＪＰＥＧ２０００符号データを、ＪＰＥＧ２０００のデコーダに渡すＪＰＥＧ２０００準拠の符号データに変換する部分について説明する。

【００２７】

また、ユーザは、Ｗｉｎｄｏｗｓ（登録商標）マシン等を使用してホームページを開き、そこに書かれているＪＰＥＧ２０００の画像へのリンクをクリックすることで、ＪＰＥＧ２０００の画像をユーザの用途に適した画像サイズや解像度で表示するために必要な断片的なデータを取得し、キャッシュする。そして、それらのキャッシュデータから一つのビットストリームを作り出し、デコードし、表示するという場合を想定する。

【００２８】

ここで、一般的なＪＰＥＧ２０００の符号データについて説明する。図３は、Layer-resolution level-component-position progression（以下、「ＳＮＲ Progression」と記す。）に沿って記録されたＪＰＥＧ２０００ファイルの構成を示す概念図である。ＳＮＲ Progressionに準じた場合、layer／resolution／component／positionの順に記録される。このような符号データの並び方を規定する

ことは、「progression order」と呼ばれる。

【００２９】

また、図４は、ＪＰＥＧ２０００の解像度スケーラビリティを説明する図である。すなわち、解像度（画像サイズ）とResolution番号との関係を示すための図である。図４では、最も小さい解像度の画像のresolution番号を０とし、resolution番号が１つ増加するごとに画像の幅と高さが２倍になっていく。また、各layer内は、resolution番号の小さい順にデータが格納されている。そして、Layer番号は、復元する画像の原画に対するＳ／Ｎ比に対応しており、layer番号が小さいほどＳ／Ｎ比が悪くなる。

【００３０】

さらに、１つのＪＰＥＧ２０００ファイル内でのresolution番号、layer番号及びcomponent番号の最大値は、エンコーダによって予め設定されており、そのパラメータに従ってエンコードされており、その情報は符号データの中に格納されている。また各packetは、そのpacket内に格納されているcode-blockの情報を管理しているpacket header部と、各code-blockの符号データとから構成されている。

【００３１】

このようなＪＰＥＧ２０００符号データを使うことによって、ユーザは、サーバにある全ての画像データを取得する必要がなく、クライアント側で表示等するために必要な部分の符号データのみをサーバから受信すればよい。尚、ユーザ側（クライアント・コンピュータ）の受信データの単位としては、ＪＰＥＧ２０００のpacket、或いはpacketよりも更に小さい符号化単位であるコードブロック単位が考えられる。本実施形態では、ユーザがサーバから受信するデータ単位としてpacket単位を想定する。

【００３２】

図５は、第１の実施形態におけるサーバとクライアント間でのpacket単位のデータのリクエスト及びレスポンスを説明するための概念図である。

【００３３】

図５において、クライアント５０１はサーバ５０２に画像のタイル番号、reso

lution levelと、layer、component、position番号を指定して必要なデータを要求する。そして、サーバ５０２は、画像５０３のコードストリームを解析して、指定された画像のタイル番号、resolution levelとlayer、component、position番号に相当するpacketデータを抜き出してクライアント５０１に送り返す。

【００３４】

次に、図６Ａ及び図６Ｂを用いて本実施形態で使用するサーバ・コンピュータで管理されているＪＰＥＧ２０００ファイルの一例について説明する。図６Ａは、サーバ・コンピュータに格納されている原画像データをタイルに分割した例を示す図である。図６Ａでは、縦横とも各々４つのタイル、すなわち画像全体で１６個のタイルに分割した例が示されている。

【００３５】

また、図６Ｂは、図６Ａで示したタイル分割された画像データに関するＪＰＥＧ２０００符号データ（すなわち、当該画像データを「ＳＮＲ Progression」でエンコードしたデータ）のデータ構造を示す図である。図６Ｂに示される符号データは、layerは０～２の３種類（Lay0、Lay1、Lay2）、resolution levelは、０～３の４種類（Res0、Res1、Res2、Res3）、componentとpositionは各１種類（Com0、Pos0）である場合のＪＰＥＧ２０００で符号化した時の符号データの構成例である。

【００３６】

例えば、ＪＰＥＧ２０００の最大解像度の画像サイズを１０２４×１０２４画素とした場合、各タイルの画像サイズは２５６×２５６画素となる。また、各タイルが４種類のresolution levelを持つ場合の各解像度での画像サイズは、２５６×２５６、１２８×１２８、６４×６４、３２×３２画素となる。

【００３７】

さらに、ＪＰＥＧ２０００符号データ内のメインヘッダ部分については、図６Ｂに示すように、画像サイズを示すパラメータであるXsiz、Ysizが共に１０２４であり、画像全体の画像サイズを示している。一方、タイルのサイズを示すXTsiz、YTsizは、上述したタイルのサイズである２５６の値を持っている。これらのパラメータにより、このＪＰＥＧ２０００符号データは、１６個のタイル

に分割された符号データを有することになる。そして、これらの各タイルの符号データが、図６Ｂに示すように「Tile part Header」で区切られて格納されている。

【００３８】

図７は、第１の実施形態におけるクライアント・コンピュータ上のアプリケーションでの処理手順の概要を説明するためのフローチャートである。本実施形態においては、クライアント側では、ＪＰＥＧ２０００画像表示アプリケーションが起動しており、図７に示されるフローチャートは、表示したいＪＰＥＧ２０００ファイルをウエブ等の別の手段で指定した後、画像を表示する部分の処理フローである。

【００３９】

まず、これからの処理を制御するための内部変数（パラメータ）を初期化する（ステップＳ７０７）。具体的には、指定されているＪＰＥＧ２０００ファイルの画像サイズや、エンコード・パラメータ等の情報をサーバに対して問い合わせ、図１３に示すこれから表示する符号化データのキャッシュを管理するための情報をメモリやファイル上に作成し、その変数（パラメータ）等を初期化する。すなわち、図１３は、クライアント側で制御している符号データのキャッシュの論理構造を説明するための図である。図１３において、１３００は、キャッシュファイル全体の論理構造であり、図に示すように、キャッシュ全体に関する画像管理情報フィールド１３０２、Main header dataフィールド１３０３、各Tileに関するデータフィールド１３０１に分けることができる。

【００４０】

ここで、Tileに関するデータフィールド１３０１には、この画像データをTile分割した時のTileの個数分のフィールドが存在する。１３０４は、Tileに関するデータのフィールド１３０１の詳細を示したものである。この中には、Tileデータ管理用データフィールド１３０４ａ、Tile headerフィールド１３０４ｂ、packetデータ管理用データ１３０４ｃに分けられる。尚、packetデータ管理用データフィールド１３０４ｃは、Tile内に存在するpacketの個数分のフィールドがある。

【００４１】

次に、Tileデータ管理用データフィールド１３０４ａの詳細を１３０５に示す。Tileデータ管理用データフィールド１３０４ａは、１３０５に示すように、Tileのデータ長１３０５ａ、Tile番号１３０５ｂ、packet数１３０５ｃ、Resolution数１３０５ｄ、Layer数１３０５ｅ、Component数１３０５ｆ、Position数１３０５ｇの各パラメータから成る。

【００４２】

さらに、packetデータの管理用データ１３０４ｃの詳細を１３０６に示す。packetデータの管理用データ１３０４ｃは、１３０６に示すように、packetのデータ長１３０６ａ、ポインタ値１３０６ｂ、packet番号１３０６ｃ、packetの基本情報１３０６ｄから成る。サーバから受信した実際のpacketのデータは、ポインタ値１３０６ｂで示すポインタの先に保存される。

【００４３】

図１３からも分かるように、packetデータをフル充填した場合は、テーブルを管理するための情報等が数多く存在するため、サーバで管理している元の符号データのバイト数よりも大きくなってしまう。すなわち、クライアント側では、このキャッシュ情報と、キャッシュ情報から１つの符号データを生成するため、クライアント側で管理する符号データ関連のバイト数は、全タイルがフル充填されたときに最大となり、その時のサイズは、サーバで管理している元の符号データのサイズの２倍以上になる。尚、これらの各Tileに関するデータ１３０４は、ファイルに格納され、１３００内の１３０１で示すフィールドには、ファイル名で該当するデータと関連付けを行っているものとする。

【００４４】

ステップＳ７０７で内部変数が初期化された後、ユーザが表示のための操作を行う（ステップＳ７００）。例えば、アプリケーション起動時の指定されたファイルの１回目の表示では、開いたウィンドウサイズに収まる画像サイズを計算して、当該ウィンドウサイズに一致する画像サイズで表示するようにする。そして、その後に本ステップＳ７００でユーザの操作を受け付けるようにしてもよい。

【００４５】

次に、上記ステップＳ７００でのユーザ操作により新たに必要になったpacketを全て算出する（ステップＳ７０１）。その後、ステップＳ７０１で算出した新規に必要となったpacketのデータについてサーバに対して要求を発行する（ステップＳ７０２）。次いで、要求した全packetのデータをサーバから受信し、クライアント・コンピュータ上にキャッシュする（ステップＳ７０３）。

【００４６】

さらに、高速読み出しが可能なようにキャッシュ形式で管理している符号データを複数のＪＰＥＧ２０００符号データ形式に変換することによって符号データの作成を行う（ステップＳ７０４）。この符号データの作成処理の詳細については後述する。その後、ステップＳ７０４で作成したＪＰＥＧ２０００符号データをデコードして表示する（ステップＳ７０５）。そして、ユーザが終了を要求したか否かを判定する（ステップＳ７０６）。その結果、ユーザが終了を要求した場合（Ｙｅｓ）、本アプリケーションを終了する。一方、ユーザが終了を要求せずに他の表示要求を行った場合（Ｎｏ）、ステップＳ７０１に戻って上記処理を実行する。

【００４７】

以下では、上記クライアント・コンピュータ上のアプリケーションでの処理手順の具体的な処理内容について説明する。

【００４８】

まず、クライアント・アプリケーションが起動した直後で、このアプリケーションの初期画像表示サイズを１２８×１２８画素とし、このサイズに入る画像データを表示することを考える。この場合、ステップＳ７０２で算出される必要なpacketは、最低解像度の画像サイズであってＳＮＲが最大のデータを表示することを考えると、図６Ｂ内のタイル０からタイル１５までの全１６タイルにおける、「Lay0/Res0/Com0/Pos0」、「Lay1/Res0/Com0/Pos0」、「Lay2/Res0/Com0/Pos0」で示す３×１６個のpacketが必要なpacketである。

【００４９】

そこで、クライアント・プログラムは、これらのpacketをサーバ・プログラムに要求する。一方、サーバ・プログラムは、この全タイルの３×１６個のpacket

を切り出す処理を行う。その後、クライアント・プログラムは、これらのpacketを受信した後デコードして表示を行う。

【００５０】

次に、図７のステップＳ７０４における符号データ作成についての詳細な説明を図８～１０を用いて行う。図８Ａは、図７におけるステップＳ７０４符号データ作成処理の概要を説明するためのフローチャートである。まず、サーバから受信した必要なpacketに関するＪＰＥＧ２０００符号データのメインヘッダを解析する（ステップＳ８００）。次に、ステップＳ８００におけるメインヘッダを解析した結果から、符号データ内のタイルの個数を計算する（ステップＳ８０１）。本実施形態においては、タイルの個数の計算は、以下に示す式（１）で行うものとする。

【００５１】

$$Xtiles = (Xsiz + XTsiz - 1) / XTsiz$$
$$Ytiles = (Ysiz + YTsiz - 1) / YTsiz \qquad (1)$$
$$Tiles = Xtiles \times Ytiles$$

そして、式（１）で求められたタイルの個数だけ、ステップＳ８０２以下の処理が行われる。

【００５２】

ステップＳ８０２は、各タイルの符号変換処理を行う部分である。この処理の詳細については後述する。そして、全タイルについての符号変換処理が終了したか否かを判断する（ステップＳ８０３）。その結果、未処理のタイルがある場合（Ｎｏ）、ステップＳ８０２に戻る。一方、全タイルについて符号変換処理が終了した場合（Ｙｅｓ）、本処理を終了してステップＳ７０５に進む。

【００５３】

図８Ｂは、ステップＳ８０２における符号変換処理を詳細に説明するためのフローチャートである。まず、今回処理するタイルがフル充填、すなわち全packetデータがキャッシュされているか否かを判断する（ステップＳ８０４）。具体的には、図１３に示すpacketのデータ長１３０６ａ、或いはpacketデータへのポインタ値１３０６ｂを見ることで判断できる。これらの値は、未受信の場合は「０

」であり、packetデータを受信すると、packetデータのバイト数や、実際に格納されているアドレス或いはファイル名等の各値が格納される。これにより、このタイル内の全てのpacketに対して、データ長１３０６ａ或いはポインタ値１３０６ｂの値を判断し、各値が「０」でなければフル充填であると判断できる。逆に、１つでも「０」の値が存在すれば、まだフル充填ではないと判断することができる。その結果、フル充填されている場合（Ｙｅｓ）、ステップＳ８０５に進み、それ以外の場合（Ｎｏ）、ステップＳ８０７に進む。

【００５４】

ステップＳ８０５では、フル充填状態の符号データを作成したか否かを判断する。その結果、まだ作成されていない場合（Ｎｏ）はステップＳ８０６に進み、作成済みの場合（Ｙｅｓ）は当該符号変換処理を終了する。ステップＳ８０６では、完全符号データを作成したことを示すフラグをセットする。当該フラグのセットに係る処理の詳細については後述する。そして、実際のキャッシュデータから符号データを作成する「符号ファイル作成」処理が行われる（ステップＳ８０７）。この符号ファイル作成処理の詳細については後述する。

【００５５】

ここで、図８のステップＳ８０６の詳細な処理内容について説明する。図１４は、図８のステップＳ８０６の完全符号データ作成済を示すフラグのセットを詳細に説明するためのフローチャートである。この処理は比較的容易であり、まず、フル充填されたタイルのTileデータ管理用データ１３０４のファイルを削除する（ステップＳ１４００）。これにより、図１３で示すTileに関するデータ１３０４、Tileデータの管理用データ１３０５、Packetデータの管理用データ１３０６の領域が削除される。次に、図１１等で示すタイル毎の符号データファイルに置き換えて完全符号データをセットする（ステップＳ１４０１）。すなわち、完全符号データ作成済を示すフラグのセットとは、図１３に示す１３００内の１３０１で示すTileに関するデータの値を変更することである。

【００５６】

上記処理を行うことにより、当該処理後にクライアントのアプリケーションがこのタイルのデータを必要とするときは、１３０１で示すフィールドに格納され

ているファイル名をアプリケーションに渡すだけでよく、データ変換等の処理は全く発生せず、高速に行うことができる。さらに、キャッシュとして保持しなければならないデータ量が、管理テーブルの形で保持する場合よりも符号データを直接持つことになるため、データ容量的に小さくて済む。

【００５７】

また、タイルがフル充填されているか否かの判断は、キャッシュとして保持されているTileのデータ長のフィールド１３０５ａを確認すればよい。図１５は、タイルがフル充填されて完全な符号データに置き換わった時の状態を示す図である。具体的な確認方法は、図１５に示されるフィールドを１６ビット長の整数のフィールドとする場合、フル充填されて置き換えた場合は、符号データの先頭１６ビットになる。この場合、符号データの先頭は、ＳＯＣマーカ・コード（０ｘＦＦ４Ｆ）であり、整数でこの値を読むと先頭のビット（１５ビット目）が”１”であるので、負の値になる。３２ビット長の整数としても結果は同じであり、ＳＯＣマーカ・コード、ＳＩＺマーカ・コードであるので、（０ｘＦＦ４ＦＦＦ５１）である。すなわち、Tileデータの管理用データの先頭を必要なビット数の整数としてリードした場合、正の整数の場合はまだ充填中であり、負の整数の場合は、フル充填で符号データに変換済みであると容易に判断することができる。

【００５８】

上述したように、１度フル充填されたキャッシュデータから符号ファイルを生成した後は、再度符号ファイル作成処理（ステップＳ８０７）を行わないように制御することにより、キャッシュが充填されて行くに連れて、符号ファイル作成処理を高速に行うことができる。

【００５９】

図９は、図８に示すステップＳ８０７の符号ファイル作成処理を詳細に説明するためのフローチャートである。まず、タイルに含まれるpacketの数をiPMaxにセットする（ステップＳ９００）。次に、「Tile part Header」を仮出力する（ステップＳ９０１）。ここで仮出力するのは、この「Tile part Header」にはタイルの符号長を格納するフィールドがあり、このタイルの符号長は以下の処理を終了した時点で計算できる値であるためである。

【００６０】

次に、変数iP、iLに初期値として「０」をセットする（ステップＳ９０２、Ｓ９０３）。ここで、変数iPはpacketのＩＤを示す変数であり、変数iLはタイルの符号長を示す変数である。次に、変数iPで示されるpacketがキャッシュされているか否かを判断する（ステップＳ９０４）。その結果、変数iPがキャッシュされている場合（Ｙｅｓ）はステップＳ９０５へ進み、変数iPがキャッシュされていない場合（Ｎｏ）はステップＳ９０７に進む。

【００６１】

ステップＳ９０７では、該当するpacketデータが無いことを示す「Zero Length Packet」のコード（以下、「ＺＬＰコード」と略す。）を出力する。そして、このＺＬＰコードのバイト数である「１」を変数iLに加える（ステップＳ９０８）。一方、ステップＳ９０５では該当するpacketのデータを出力する。そして、そのバイト数を変数iLに足し込んで更新する（ステップＳ９０６）。

【００６２】

ステップＳ９０６及びＳ９０８の処理の後、変数iPをインクリメントする（ステップＳ９０９）。そして、全packetについて処理したかを判断する（ステップＳ９１０）。その結果、まだ全packetを処理していない場合（Ｎｏ）、ステップＳ９０４に戻って上記処理を行う。一方、全packetを処理した場合（Ｙｅｓ）、変数iLから「Tile part Header」である「ＳＯＴマーカ」を更新して出力し（ステップＳ９１１）、当該符号ファイル生成処理を終了する。そして、前述したように当該符号ファイルを汎用のＪＰＥＧ２０００デコーダでデコードし、デコードされた画像データを表示する（ステップＳ７０５）。

【００６３】

すなわち、上述したように、本実施形態に係るクライアントは、サーバが管理する符号データのうち、断片的な第１の符号データを格納した格納手段（例えば、ハードディスク２０６）を備え、ＪＰＥＧ２０００符号データを作成する符号データ作成装置として機能する。本実施形態に係るクライアントでは、まず、当該クライアントにおいてＪＰＥＧ２０００符号データの作成に必要となる符号データと、予めハードディスク２０６等の格納手段に格納された第１の符号データ

（例えば、現在表示部２０３に表示されている画像データについての符号データ）とから、不足する第２の符号データ（例えば、表示部２０３で拡大表示するために必要な画像データについての符号データであって、第１の符号データには含まれていないもの）を算出する。そして、サーバに対して、算出された第２の符号データを要求し、サーバから、第２の符号データを取得し、取得された第２の符号データをハードディスク２０６等の格納手段に格納する。次いで、取得された第２の符号データのヘッダ情報を解析して、対象となる符号データを複数の独立した符号データに分割する。さらに、分割された単位毎に、独立した符号データを構成する全ての符号データがハードディスク２０６等の格納手段に格納されているか否かを判定する。そして、独立した符号データを構成する全ての符号データが格納されていない場合、格納されていない部分にダミー符号データを格納し、格納された符号データをＪＰＥＧ２０００符号データとして出力することを特徴とする。

【００６４】

また、上記クライアント（符号データ作成装置）では、符号データが、パケット単位で取り扱われることを特徴とする。さらに、上記クライアント（符号データ作成装置）では、ダミー符号データが、ＪＰＥＧ２０００で規定されているzero length packetデータであることを特徴とする。さらにまた、上記サーバとクライアントが、ネットワークを介して互いに通信可能であることを特徴とする。

【００６５】

さらに、上記クライアント（符号データ作成装置）は、画像データを表示する表示手段（例えば、表示部２０３）をさらに備え、上記第１の符号データが当該画像データの符号データであって、表示部２０３等の表示手段に表示された画像データの表示領域の移動又は拡大表示に応じて、上記第２の符号データの算出において必要となる符号データを設定する。また、上記クライアント（符号データ作成装置）は、出力された上記ＪＰＥＧ２０００符号データをしてデコードし、デコードされた画像データを表示部２０３等の表示手段に画面表示することを特徴とする。

【００６６】

ここで、具体例を用いて、上記フローチャートで示した符号ファイル作成処理を説明する。図１０は、本発明の第１の実施形態における初期画面での符号ファイル処理後の各タイルの符号ファイルの構成を示す図である。ここで、クライアント・コンピュータから要求されるpacketは、前述のとおり全タイルの「Lay0/Res0/Com0/Pos0」、「Lay1/Res0/Com0/Pos0」、「Lay2/Res0/Com0/Pos0」である。

【００６７】

従って、図１０に示すように、全１６個のタイルに対して、符号１００１で示すように上記３つのpacketデータ以外のpacketについてはＺＬＰが格納された符号データが作成される。また、各タイル毎に作成される符号データのメインヘッダ部分は、メインヘッダ内のＳＩＺマーカコード内のXsiz、Ysizの各フィールドがタイルのサイズである２５６に変更されている。これにより、各タイル毎に独立した符号ファイルを作成することができる。

【００６８】

すなわち、本実施形態に係るクライアント（符号データ作成装置）では、符号データ（画像データ）を所定サイズのタイル単位（例えば、式（１）に記載の方法で算出された単位）で分割することを特徴とする。また、同クライアントでは、独立した符号データのそれぞれのヘッダ情報を、分割されたタイル単位のサイズに変更することを特徴とする。

【００６９】

次に、ステップＳ７０１でユーザから拡大表示が指示された場合について説明する。この拡大表示では、まず初期画面より２倍に拡大するものとする。この場合、表示ウィンドウサイズは１２８×１２８画素であるとすると、表示できるタイルは４個になる。例えば、図６Ａに示すTile5、Tile6、Tile9、Tile10の４個のタイルのResolution１（画像サイズ：６４×６４画素）を表示するものとする。この場合、ステップＳ７０２では、Tile5、Tile6、Tile9、Tile10の４つのタイルの「Lay0/Res1/Com0/Pos0」、「Lay1/Res1/Com0/Pos0」、「Lay2/Res1/Com0/Pos0」がクライアント･コンピュータからサーバに対して要求される。

【００７０】

さらに、ユーザから２倍に拡大する要求が発行された場合、表示ウィンドウサイズを１２８×１２８画素から２５６×２５６画素に変更すると、上記４タイルに対してさらに、「Lay0/Res2/Com0/Pos0」、「Lay1/Res2/Com0/Pos0」、「Lay2/Res2/Com0/Pos0」が要求される。

【００７１】

その後、さらに、２倍に拡大する要求がユーザからあったとする。この場合、表示領域のウィンドウサイズを変更しないとすると、タイルが１つ分しか表示することができない。そこで、Tile5のみを表示するとすると、Tile5の「Lay0/Res3/Com0/Pos0」、「Lay1/Res3/Com0/Pos0」、「Lay2/Res3/Com0/Pos0」をサーバに要求し表示することになる。これにより、Tile5は完全にキャッシュが充填されたことになる。図１１は、第１の実施形態における符号データの各タイル毎の符号ファイルの状態の一例を説明するための図である。

【００７２】

図１１に示すように、Tile0、Tile1、Tile2、Tile3、Tile4、Tile7、Tile8、Tile11、Tile12、Tile13、Tile14及びTile15の各タイルの符号ファイルは、図１０に示したものと同様に符号１００１で示す状態になっている。また、Tile6、Tile9及びTile10の各タイルは符号１１０１に示す状態に、Tile5は符号１１０２に示す状態になっている。この状態になるまでは、ステップＳ８０５における完全符号データを作成済みか否かの判断により全てステップＳ８０６に進むようになるが、Tile5の符号変換処理後はステップＳ８０５の判断後、ステップＳ８０６には進まずに処理を終了するようになる。これにより、Tile5については、これ以降、符号変換処理を行う必要がなくなる。これにより、キャッシュが充填されるほど符号ファイル作成処理を高速にすることができる。すなわち、本実施形態によれば、デコード／表示する領域をタイル単位で行う場合、デコード／表示に必要なタイルのみに対して処理を行うことができる。

【００７３】

また、従来は、キャッシュファイルから１つのＪＰＥＧ２０００符号データファイルを生成する場合、表示するタイル毎のマルチスレッド化は困難であった。これは、１つの符号データファイルを生成するため、ここで処理をシリアライズ

することが必須であったためである。しかし、本実施形態のように各タイル毎の符号データファイルを生成することにより、表示するタイル毎に必要なpacket要求からキャッシュファイルの生成、キャッシュファイルから符号データの作成、符号データのデコード及び表示までを各タイル毎スレッドとしてマルチスレッド化することができる。従って、これまでのシングルスレッドの場合に比べて、マルチスレッド化することによる高速化も図ることができる。

【００７４】

さらに、ファイルＩ／Ｏの処理時間がかかるようなシステムにおいては、１つの大きな符号データファイルを作成する代わりに、小さな複数の符号ファイルを生成することで、ファイルＩ／Ｏによる時間のロスを最小限にすることも可能となる。

【００７５】

＜第２の実施形態＞

上述した第１の実施形態では、キャッシュされている全てのタイルに対して符号変換処理を行うことを説明したが、本実施形態では、表示に必要な部分のタイルのみ符号変換処理を行うことにより、第１の実施形態よりもさらに高速に符号変換処理を行う場合について説明する。

【００７６】

図１２は、本発明の第２の実施形態に係る符号データ作成処理の概要を説明するためのフローチャートである。尚、上述した第１の実施形態における図８Ａのフローチャートと同じ処理の部分には同一の番号を付けている。ここでは、第１の実施形態と異なる部分のみを説明する。

【００７７】

ステップＳ８０１のタイル個数の計算の後、ステップＳ７０１でユーザが操作したコマンドを解析する（ステップＳ１２００）。そして、この解析結果を使用して、これから処理しようとしているタイルが今回の表示で表示されるタイルか否かを判断する（ステップＳ１２０１）。その結果、ここで表示に使用しないタイルであると判断された場合（Ｎｏ）、ステップＳ８０３に進む。一方、表示に使用するタイルであると判断された場合（Ｙｅｓ）、ステップＳ１２０２に進む

。

【００７８】

ステップＳ１２０２では、ステップＳ７０３で新たにpacketを要求した場合に、今回処理しようとしているタイルのpacketを取得したか否かを判断する。その結果、新規packetを取得している場合（Ｙｅｓ）はステップＳ８０２に進み、新規packetを取得していない場合（Ｎｏ）はステップＳ８０３に進む。尚、これ以降の処理手順については、第１の実施形態で説明した手順と同様である。これらの処理により、表示に必要なタイルであって、まだ完全に充填されていないタイルのみ符号変換処理を行うことができるため、当該符号化処理を第１の実施形態よりも高速に行うことができる。

【００７９】

＜第３の実施形態＞

以下、本発明の第３の実施形態に係る画像データ送信装置について、図面を参照して、詳細に説明する。本実施形態では、キャッシュから１本の符号データを作成する方法について説明する。尚、本実施形態に係るサーバ及びクライアントを含むネットワークシステムの構成は図１と同様であり、各コンピュータシステムのハードウェア構成も図２と同様である。また、クライアントコンピュータ上のアプリケーションでの処理手順についても図７に示すフローチャートと同様である。

【００８０】

図１６は、キャッシュから１本の符号データファイルを作成する場合の処理手順を説明するためのフローチャートである。図１６に示すように、まず、キャッシュ内のフィールド１３０３で保持している情報をＪＰＥＧ２０００符号データ形式に則った形式に変換し、出力ファイルにメインヘッダ（Main Header）を出力する（ステップＳ１６００）。次に、メインヘッダを解析した結果から、符号データ内のタイルの個数を計算する（ステップＳ１６０１）。尚、この際の計算式は、図８のステップＳ８０１の場合と同様である。そして、以下では、前述した式（１）で求められたタイルの個数だけ、ステップＳ１６０２以下の処理が行われる。

【００８１】

そして、各タイルごとに、当該タイルがフル充填しており、タイルの完全な符号データがあるか否かを判断する（ステップＳ１６０２）。尚、この判断方法は、前述の通りである。その結果、フル充填されていると判断された場合（Ｙｅｓ）、ステップＳ１６０４へ進み、フル充填されていない場合（Ｎｏ）、ステップＳ１６０３へ進む。

【００８２】

ステップ１６０３では、図９に示す符号ファイルの作成処理手順と同様にして符号ファイルを作成する。そして、この処理後はステップＳ１６０６へ移り、符号データを出力する。一方、ステップＳ１６０４以降では、タイルの完全な符号データからの変換が行われる。そこで、まず、メインヘッダ部分を読み飛ばす処理が行われる（ステップＳ１６０４）。この処理の具体例としては、例えば、図１５に示す符号データの先頭からＳＯＴマーカ・コードまでを読み飛ばせばよい。次に、タイルパートヘッダ（Tile part Header）内のタイルインデックス（Tile Index）の値を当該タイルのインデックス（Index）番号に置き換える（ステップＳ１６０５）。さらに、タイルパートヘッダ以降ＥＯＣマーカ・コードまで（但し、ＥＯＣマーカ・コードは含まない）をコピーする（ステップＳ１６０６）。そして、全タイルを処理したかどうかを判断し（ステップＳ１６０７）、未処理の場合（Ｎｏ）はステップＳ１６０２に戻って上記処理を繰り返す。一方、全タイルが処理された場合（Ｙｅｓ）は終了する。

【００８３】

上述した処理によって、本実施形態に係る形式でのキャッシュファイルから、１本の符号データを簡単に、且つ、データ変換等の処理がないため高速に作成することができる。

【００８４】

＜その他の実施形態＞

一般に、キャッシュファイルから１つのＪＰＥＧ２０００符号データファイルを生成する場合には、表示するタイル毎のマルチスレッド化は困難であった。なぜならば、１つの符号データファイルを生成するために、処理をシリアライズす

ることが必須であったからである。しかし、上述したように、各タイル毎の符号データファイルを生成することにより、表示のタイル毎に必要なpacket要求から、キャッシュファイルの生成、キャッシュファイルからの符号データの作成、符号データのデコード、表示までを各タイル毎スレッドとしてマルチスレッド化を行うことができる。そのため、これまでのシングルスレッドの場合に比べて、マルチスレッド化することによる高速化も図ることができる。

【００８５】

尚、本発明は、複数の機器（例えば、ホストコンピュータ、インタフェース機器、リーダ、プリンタ等）から構成されるシステムに適用しても、一つの機器からなる装置（例えば、複写機、ファクシミリ装置等）に適用してもよい。

【００８６】

また、本発明の目的は、前述した実施形態の機能を実現するソフトウェアのプログラムコードを記録した記録媒体（または記憶媒体）を、システムあるいは装置に供給し、そのシステムあるいは装置のコンピュータ（またはＣＰＵやＭＰＵ）が記録媒体に格納されたプログラムコードを読み出し実行することによっても、達成されることは言うまでもない。この場合、記録媒体から読み出されたプログラムコード自体が前述した実施形態の機能を実現することになり、そのプログラムコードを記録した記録媒体は本発明を構成することになる。また、コンピュータが読み出したプログラムコードを実行することにより、前述した実施形態の機能が実現されるだけでなく、そのプログラムコードの指示に基づき、コンピュータ上で稼働しているオペレーティングシステム（ＯＳ）などが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【００８７】

さらに、記録媒体から読み出されたプログラムコードが、コンピュータに挿入された機能拡張カードやコンピュータに接続された機能拡張ユニットに備わるメモリに書き込まれた後、そのプログラムコードの指示に基づき、その機能拡張カードや機能拡張ユニットに備わるＣＰＵなどが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれること

は言うまでもない。

【００８８】

本発明を上記記録媒体に適用する場合、その記録媒体には、先に説明したフローチャートに対応するプログラムコードが格納されることになる。

【００８９】

以上説明したように、本発明による上記実施形態で説明したファイル作成方法を用いれば、ＩＩＰを用いて送られてくるＪＰＥＧ２０００符号化データの断片的なデータをキャッシュし、このキャッシュファイルから通常のＪＰＥＧ２０００デコーダで処理できるＪＰＥＧ２０００符号化データを作る時に、今回のデコード及び表示に必要なタイル毎に独立したＪＰＥＧ２０００符号データファイルを生成することで、デコード／表示までの処理を高速に行うことができる。

【００９０】

また、キャッシュを構成するための、或いはキャッシュから符号データを生成するためのデータ容量或いはファイル容量を最小限に抑えることが可能になる。さらに、キャッシュから符号データを生成する時も、データ変換等の処理を省略することができるため、高速に生成することができる。

【００９１】

また、デコード／表示する領域をタイル単位で行う時、デコード／表示に必要なタイルのみに対して処理を行うことができる。さらに、マルチスレッド化による高速化も行うことができる。さらにまた、ファイルＩ／Ｏの処理時間がかかるようなシステムにおいては、１つの大きな符号データファイルを作る代わりに、小さな複数の符号ファイルを生成することで、ファイルＩ／Ｏによる時間のロスを最小限にすることも可能となる。あるいは、１つの符号データを作成するときでも、容易に変換を行うことができる。

【００９２】

【発明の効果】

以上説明したように、本発明によれば、クライアントにキャッシュされている断片化された符号データと、サーバから必要な分だけ受信した断片化された符号データとを用いて、クライアントにおいて汎用ＪＰＥＧ２０００デコーダで使用

可能な符号データであって、当該符号データのデコード及び画像データの表示処理を高速に行うことができる符号データを好適に作成することができる。

【図面の簡単な説明】

【図１】

本発明の第１の実施形態に係るサーバ及びクライアントを含むネットワークシステムの構成を示す概要図である。

【図２】

図１のサーバ・コンピュータ１０１、１０３又はクライアント・コンピュータ１０２ａ、１０２ｂの各コンピュータシステムのハードウェア構成の一例を示す図である。

【図３】

Layer-resolution level-component-position progression（ＳＮＲ Progression）に沿って記録されたＪＰＥＧ２０００ファイルの構成を示す概念図である。

【図４】

ＪＰＥＧ２０００の解像度スケーラビリティを説明する図である。

【図５】

第１の実施形態におけるサーバとクライアント間でのpacket単位のデータのリクエスト及びレスポンスを説明するための概念図である。

【図６Ａ】

サーバ・コンピュータに格納されている原画像データをタイルに分割した例を示す図である。

【図６Ｂ】

図６Ａで示したタイル分割された画像データに関するＪＰＥＧ２０００符号データのデータ構造を示す図である。

【図７】

第１の実施形態におけるクライアント・コンピュータ上のアプリケーションでの処理手順の概要を説明するためのフローチャートである。

【図８Ａ】

図７におけるステップＳ７０４符号データ作成処理の概要を説明するためのフローチャートである。

【図８Ｂ】

ステップＳ８０２における符号変換処理を詳細に説明するためのフローチャートである。

【図９】

図８に示すステップＳ８０７の符号ファイル作成処理を詳細に説明するためのフローチャートである。

【図１０】

本発明の第１の実施形態における初期画面での符号ファイル処理後の各タイルの符号ファイルの構成を示す図である。

【図１１】

第１の実施形態における符号データの各タイル毎の符号ファイルの状態の一例を説明するための図である。

【図１２】

本発明の第２の実施形態に係る符号データ作成処理の概要を説明するためのフローチャートである。

【図１３】

クライアント側で制御している符号データのキャッシュの論理構造を説明するための図である。

【図１４】

図８のステップＳ８０６の完全符号データ作成済を示すフラグのセットを詳細に説明するためのフローチャートである。

【図１５】

タイルがフル充填されて完全な符号データに置き換わった時の状態を示す図である。

【図１６】

キャッシュから１本の符号データファイルを作成する場合の処理手順を説明するためのフローチャートである。

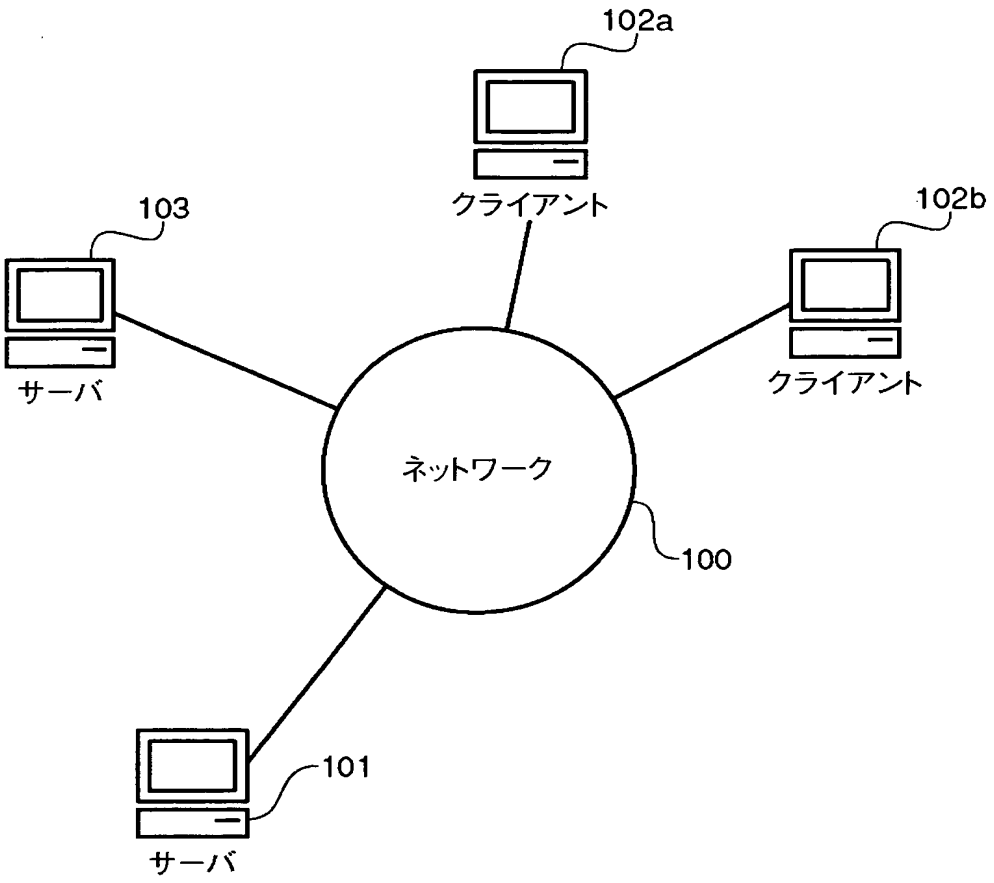
【符号の説明】

５０１　サーバ

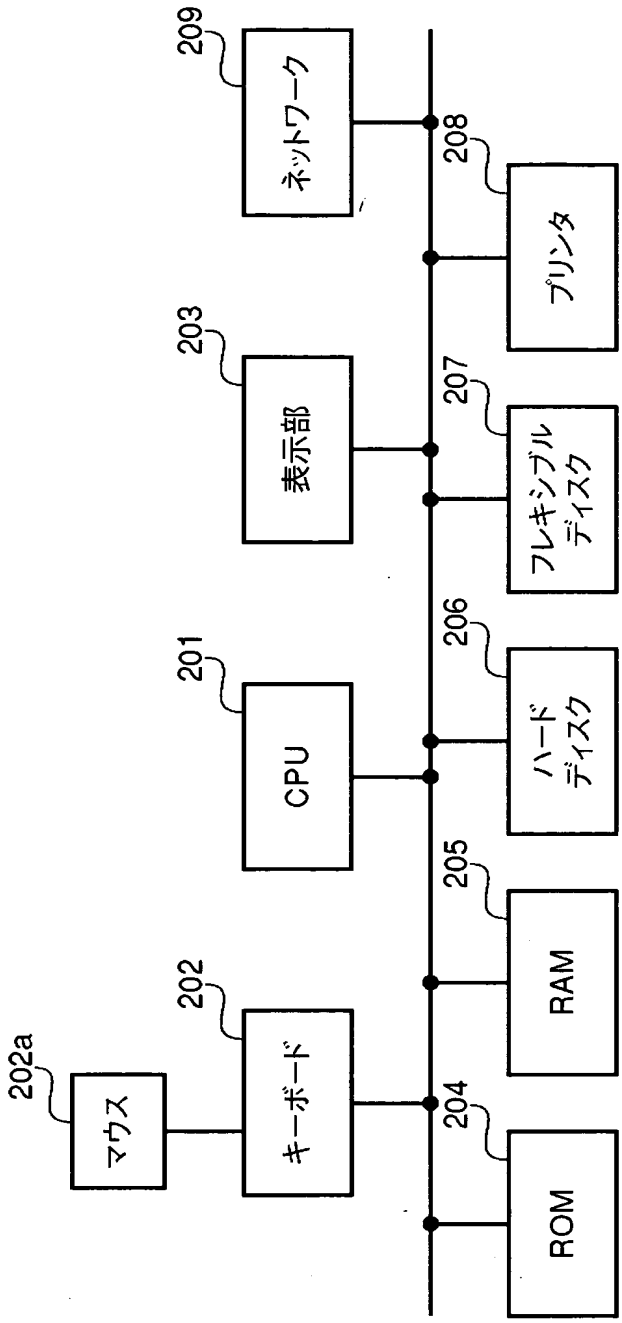
５０２　クライアント

【書類名】　図面

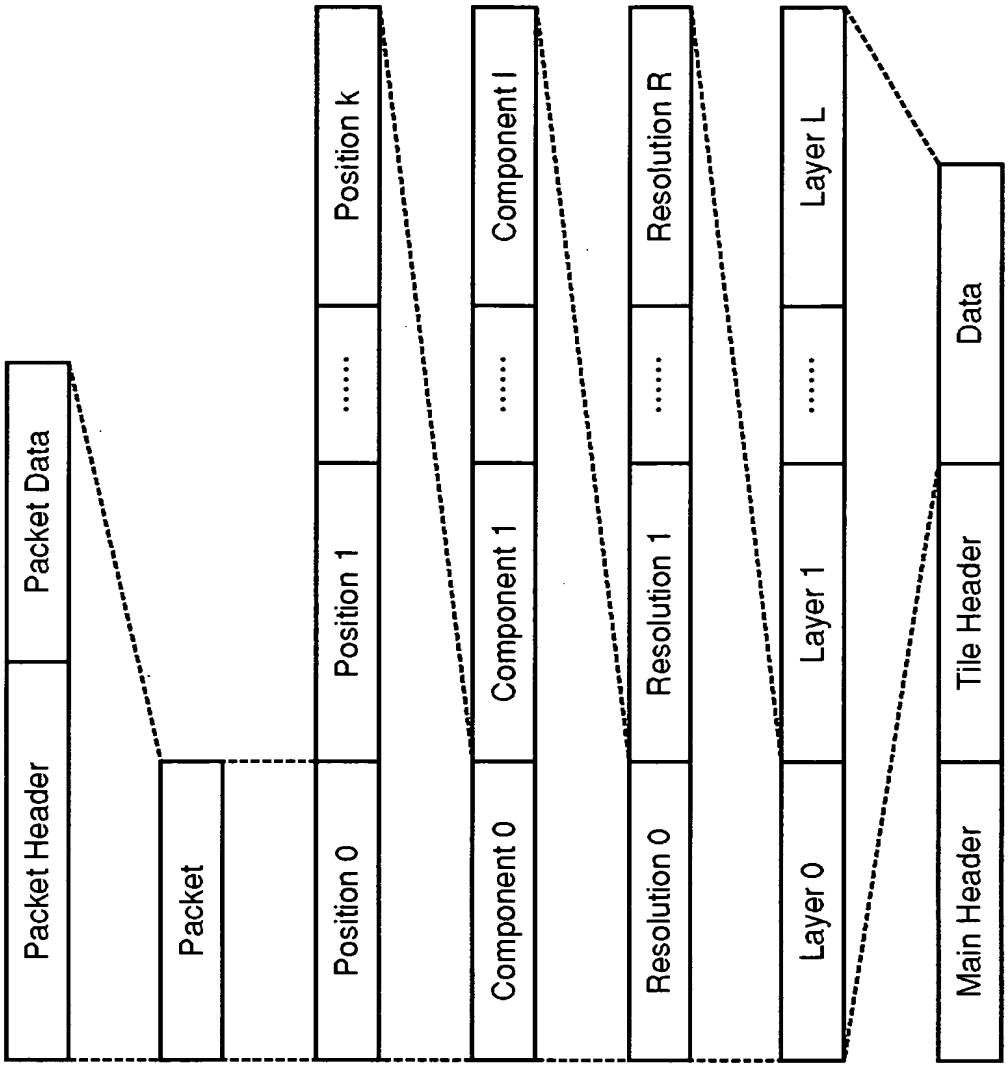
【図１】



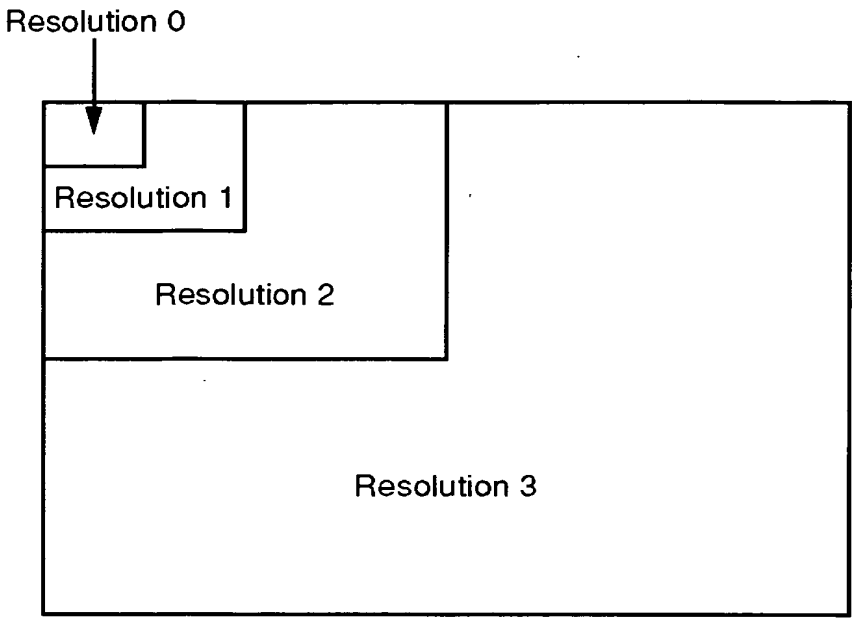
【図２】


209
ネットワーク
208
プリンタ
203
表示部
207
フレキシブル
ディスク
201
CPU
206
ハード
ディスク
205
RAM
202
キーボード
202a
マウス
204
ROM

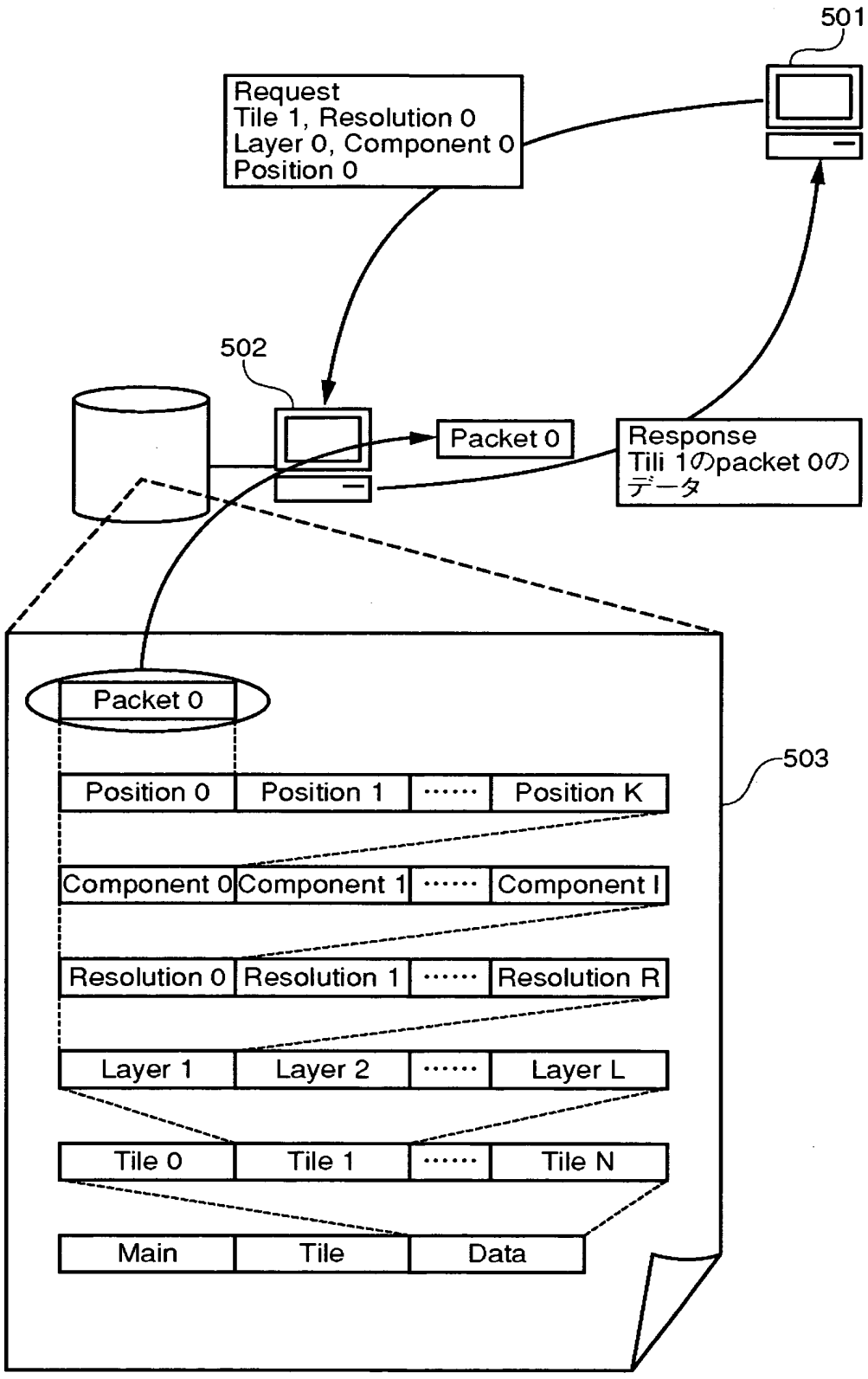
【図３】


Packet Header
Packet Data
Packet
Position 0
Position 1
.....
Position k
Component 0
Component 1
.....
Component I
Resolution 0
Resolution 1
.....
Resolution R
Layer 0
Layer 1
.....
Layer L
Main Header
Tile Header
Data

【図４】



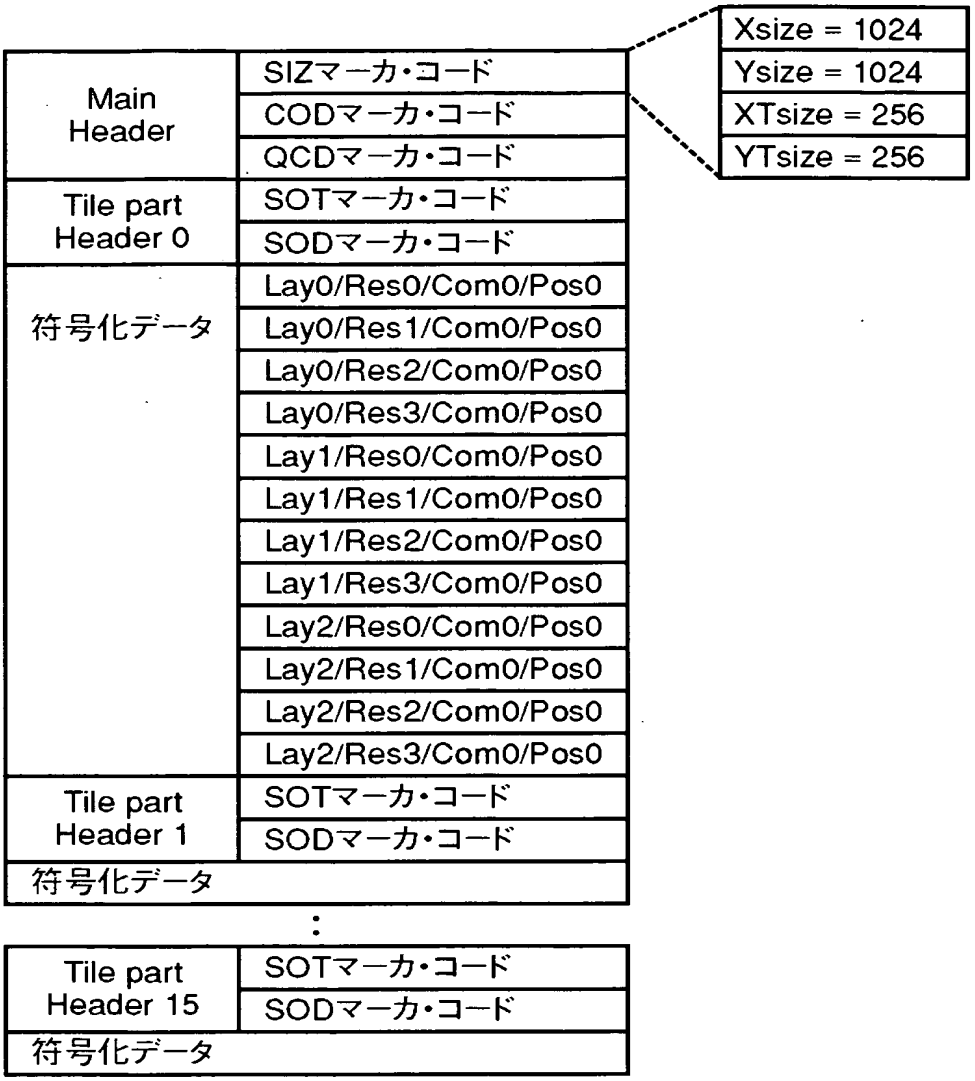
【図５】


501
Request
Tile 1, Resolution 0
Layer 0, Component 0
Position 0
502
Packet 0
Response
Tili 1のpacket 0の
データ
Packet 0
503
Position 0
Position 1
·····
Position K
Component 0
Component 1
·····
Component I
Resolution 0
Resolution 1
·····
Resolution R
Layer 1
Layer 2
·····
Layer L
Tile 0
Tile 1
·····
Tile N
Main
Tile
Data

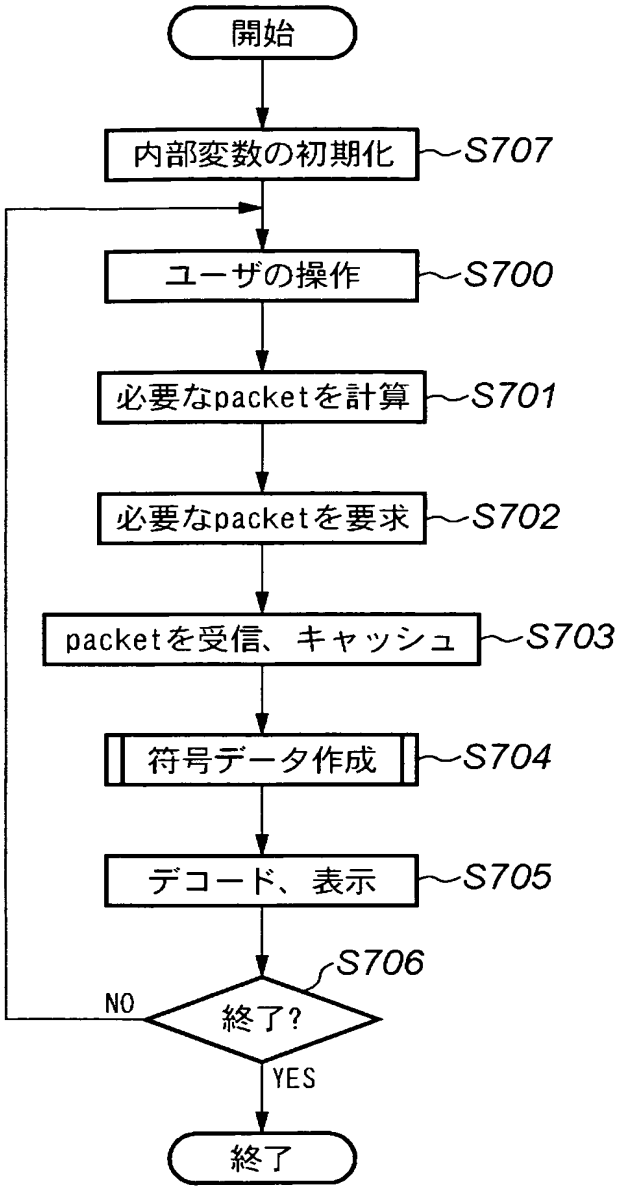
【図６Ａ】

Tile 0	Tile 1	Tile 2	Tile 3
Tile 4	Tile 5	Tile 6	Tile 7
Tile 8	Tile 9	Tile 10	Tile 11
Tile 12	Tile 13	Tile 14	Tile 15

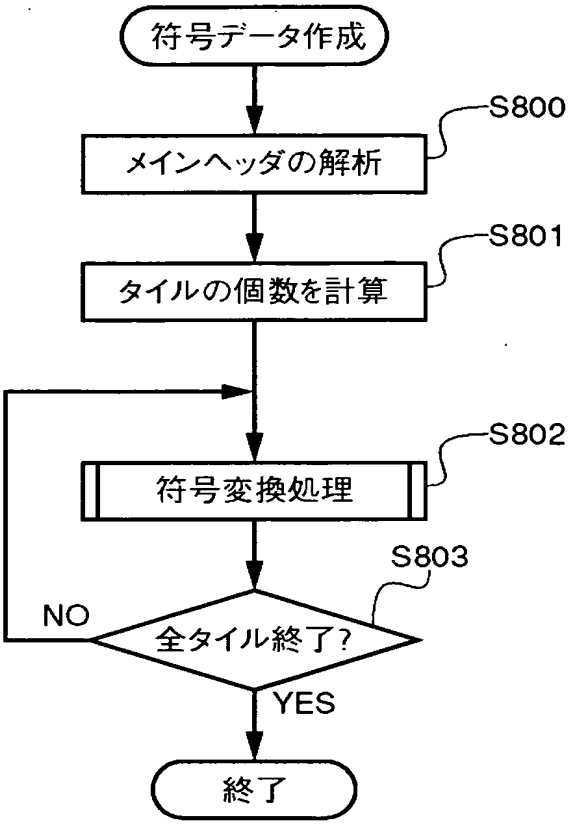
【図６Ｂ】


Main Header
SIZマーカ・コード
CODマーカ・コード
QCDマーカ・コード
Xsize = 1024
Ysize = 1024
XTsize = 256
YTsize = 256
Tile part Header 0
SOTマーカ・コード
SODマーカ・コード
符号化データ
Lay0/Res0/Com0/Pos0
Lay0/Res1/Com0/Pos0
Lay0/Res2/Com0/Pos0
Lay0/Res3/Com0/Pos0
Lay1/Res0/Com0/Pos0
Lay1/Res1/Com0/Pos0
Lay1/Res2/Com0/Pos0
Lay1/Res3/Com0/Pos0
Lay2/Res0/Com0/Pos0
Lay2/Res1/Com0/Pos0
Lay2/Res2/Com0/Pos0
Lay2/Res3/Com0/Pos0
Tile part Header 1
SOTマーカ・コード
SODマーカ・コード
符号化データ
Tile part Header 15
SOTマーカ・コード
SODマーカ・コード
符号化データ

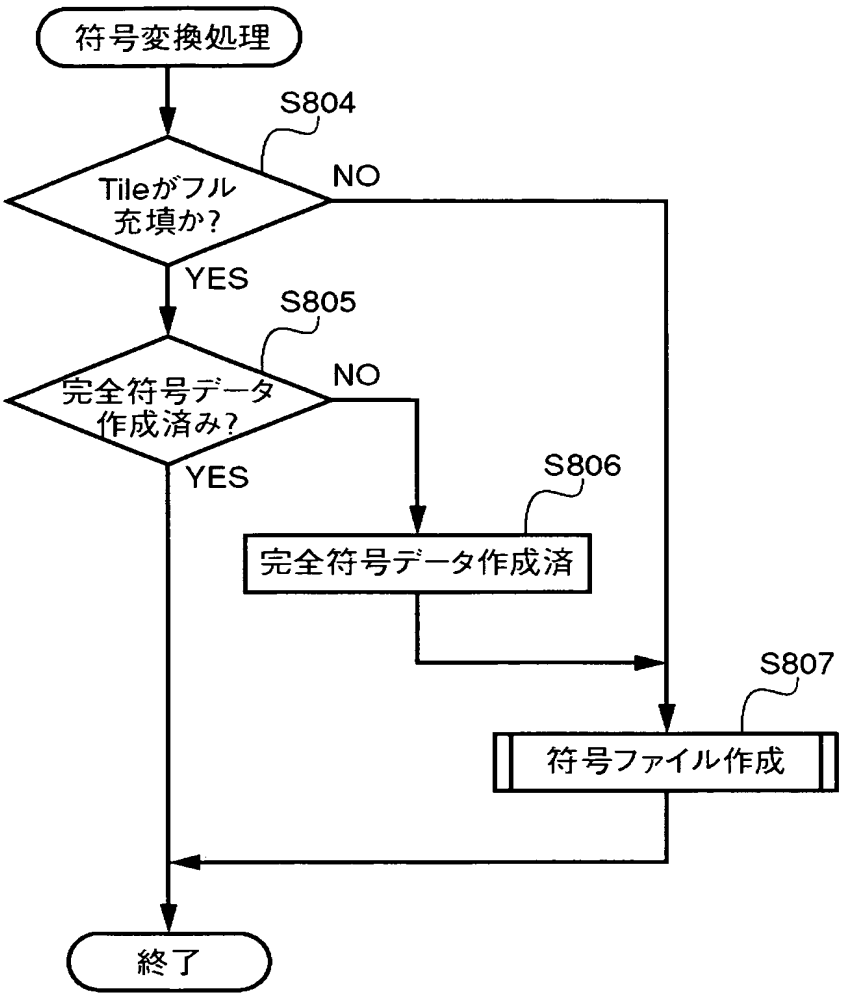
【図７】


開始
内部変数の初期化
S707
ユーザの操作
S700
必要なpacketを計算
S701
必要なpacketを要求
S702
packetを受信、キャッシュ
S703
符号データ作成
S704
デコード、表示
S705
S706
終了?
NO
YES
終了

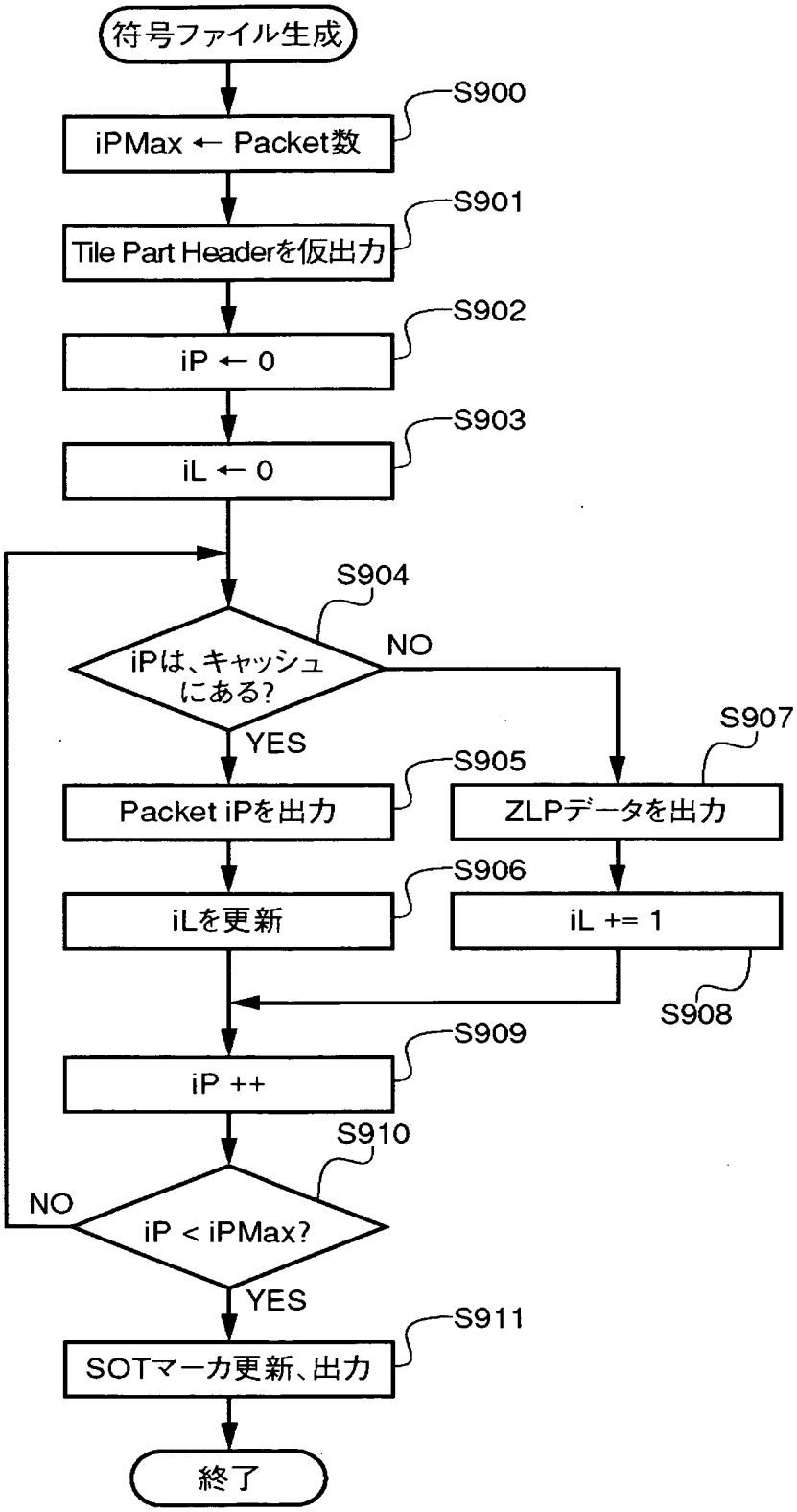
【図８Ａ】


符号データ作成
S800
メインヘッダの解析
S801
タイルの個数を計算
S802
符号変換処理
S803
NO
全タイル終了?
YES
終了

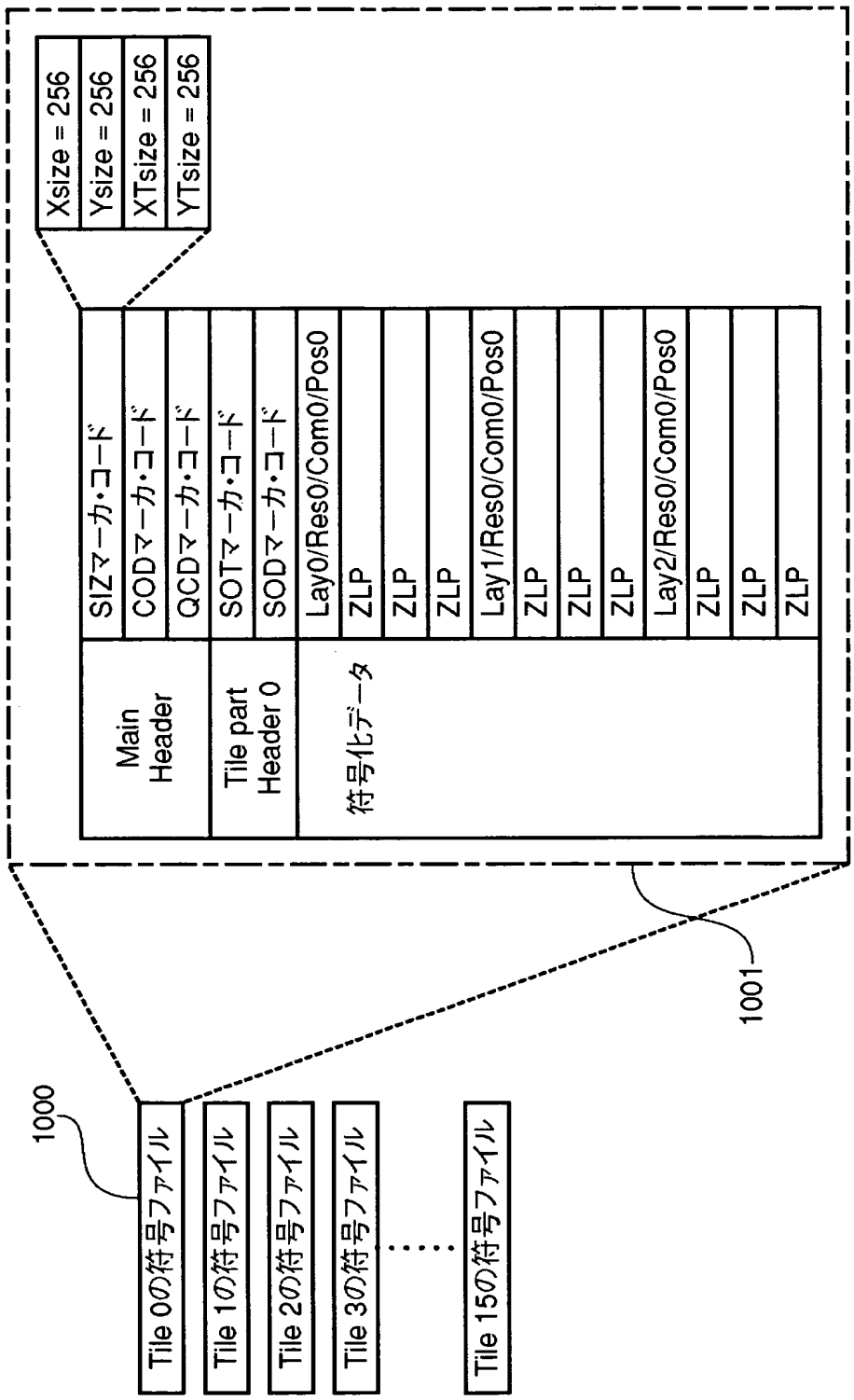
【図８Ｂ】


符号変換処理
S804
Tileがフル充填か?
NO
YES
S805
完全符号データ作成済み?
NO
YES
S806
完全符号データ作成済
S807
符号ファイル作成
終了

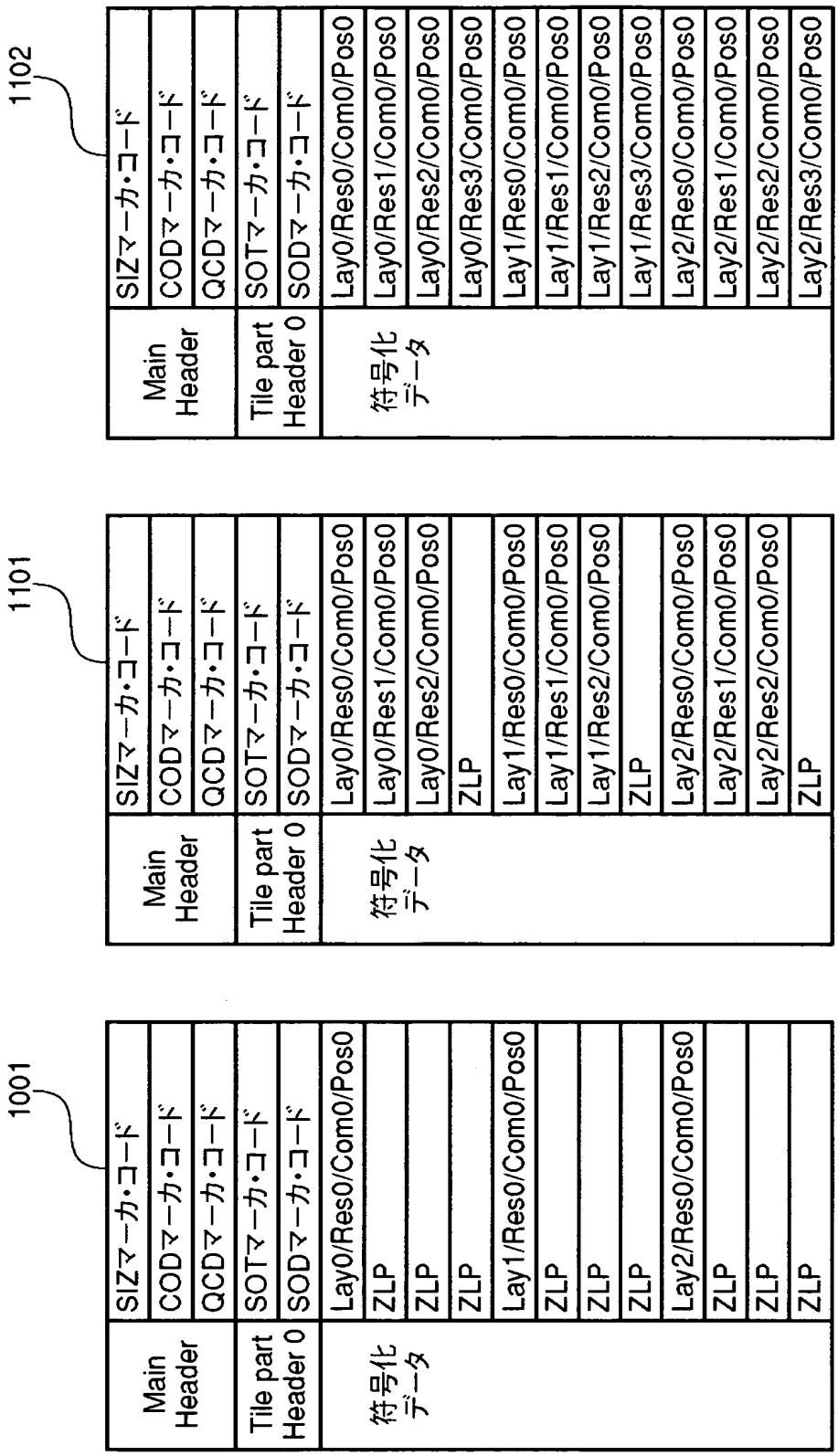
【図９】


符号ファイル生成
S900
iPMax ← Packet数
S901
Tile Part Headerを仮出力
S902
iP ← 0
S903
iL ← 0
S904
iPは、キャッシュにある?
NO
YES
S907
S905
Packet iPを出力
ZLPデータを出力
S906
iLを更新
iL += 1
S908
S909
iP ++
S910
NO
iP < iPMax?
YES
S911
SOTマーカ更新、出力
終了

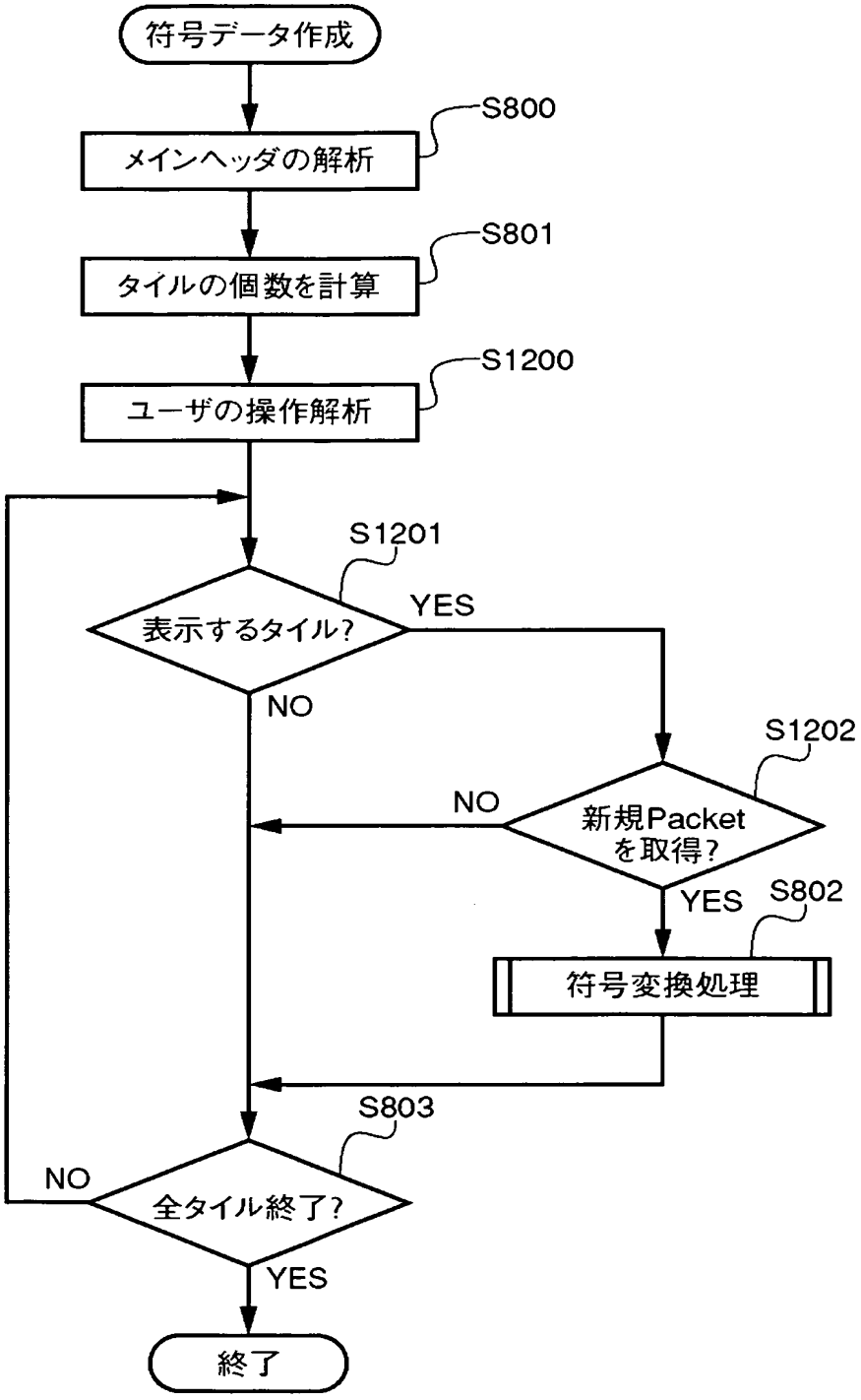
【図１０】


Xsize = 256
Ysize = 256
XTsize = 256
YTsize = 256
SIZマーカ・コード
CODマーカ・コード
QCDマーカ・コード
SOTマーカ・コード
SODマーカ・コード
Lay0/Res0/Com0/Pos0
ZLP
ZLP
ZLP
Lay1/Res0/Com0/Pos0
ZLP
ZLP
ZLP
Lay2/Res0/Com0/Pos0
ZLP
ZLP
ZLP
Main Header
Tile part Header 0
符号化データ
1001
1000
Tile 0の符号ファイル
Tile 1の符号ファイル
Tile 2の符号ファイル
Tile 3の符号ファイル
Tile 15の符号ファイル

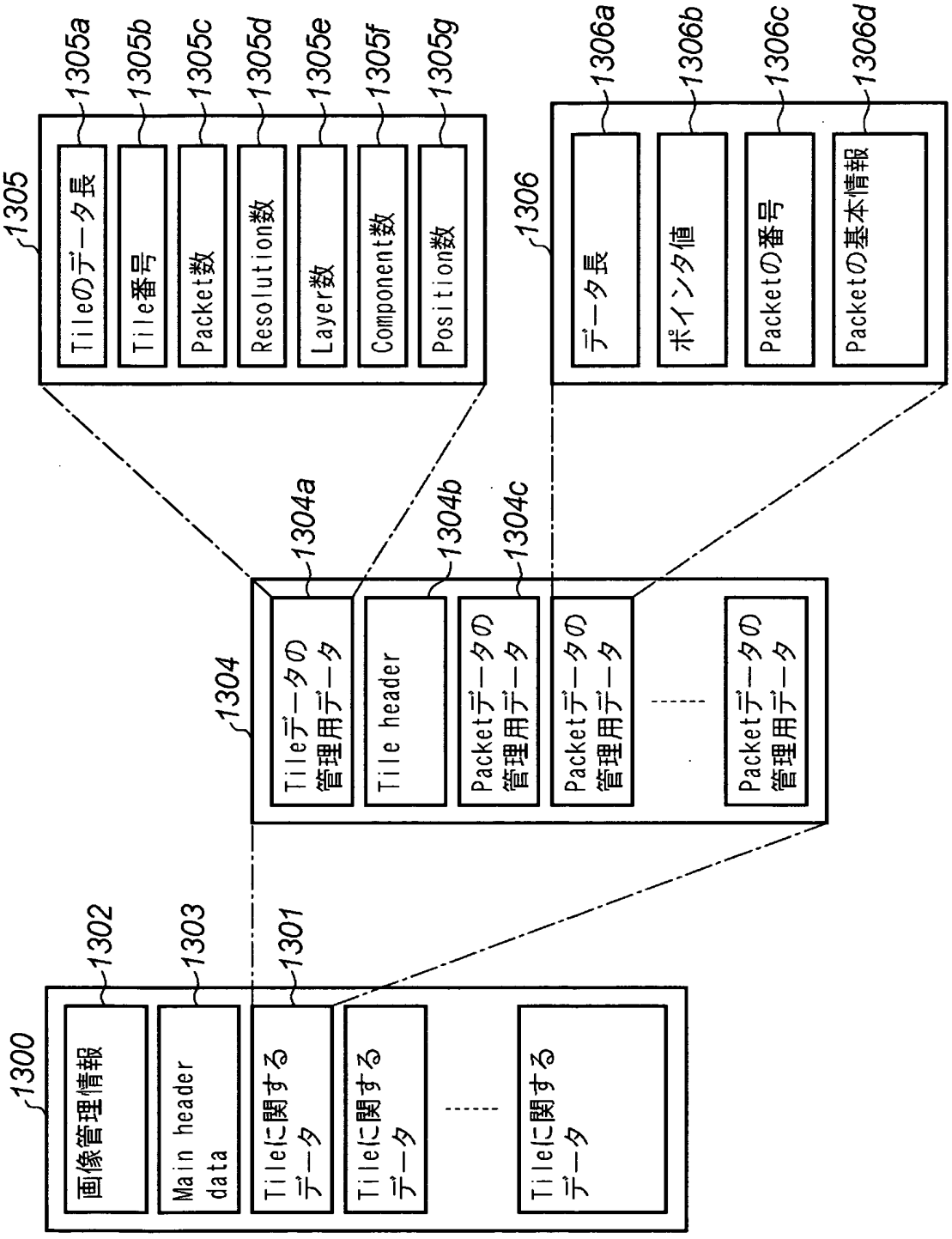
【図１１】


1001
1101
1102
Main Header
Tile part Header 0
符号化データ
SIZマーカ・コード
CODマーカ・コード
QCDマーカ・コード
SOTマーカ・コード
SODマーカ・コード
Lay0/Res0/Com0/Pos0
ZLP
Lay1/Res0/Com0/Pos0
Lay2/Res0/Com0/Pos0
Lay0/Res1/Com0/Pos0
Lay1/Res1/Com0/Pos0
Lay2/Res1/Com0/Pos0
Lay0/Res2/Com0/Pos0
Lay1/Res2/Com0/Pos0
Lay2/Res2/Com0/Pos0
Lay0/Res3/Com0/Pos0
Lay1/Res3/Com0/Pos0
Lay2/Res3/Com0/Pos0

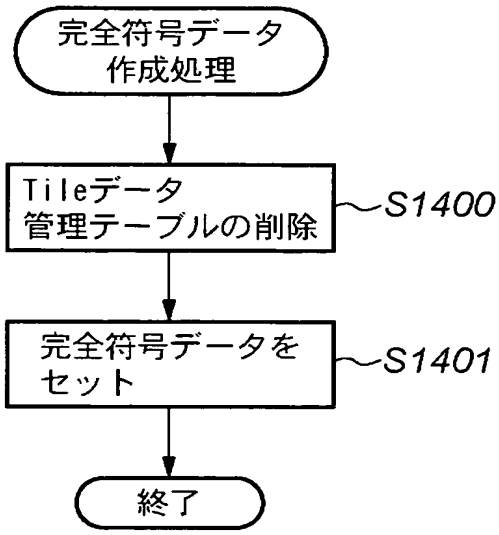
【図１２】


符号データ作成
S800
メインヘッダの解析
S801
タイルの個数を計算
S1200
ユーザの操作解析
S1201
表示するタイル?
YES
NO
S1202
新規Packet
を取得?
NO
YES
S802
符号変換処理
S803
全タイル終了?
NO
YES
終了

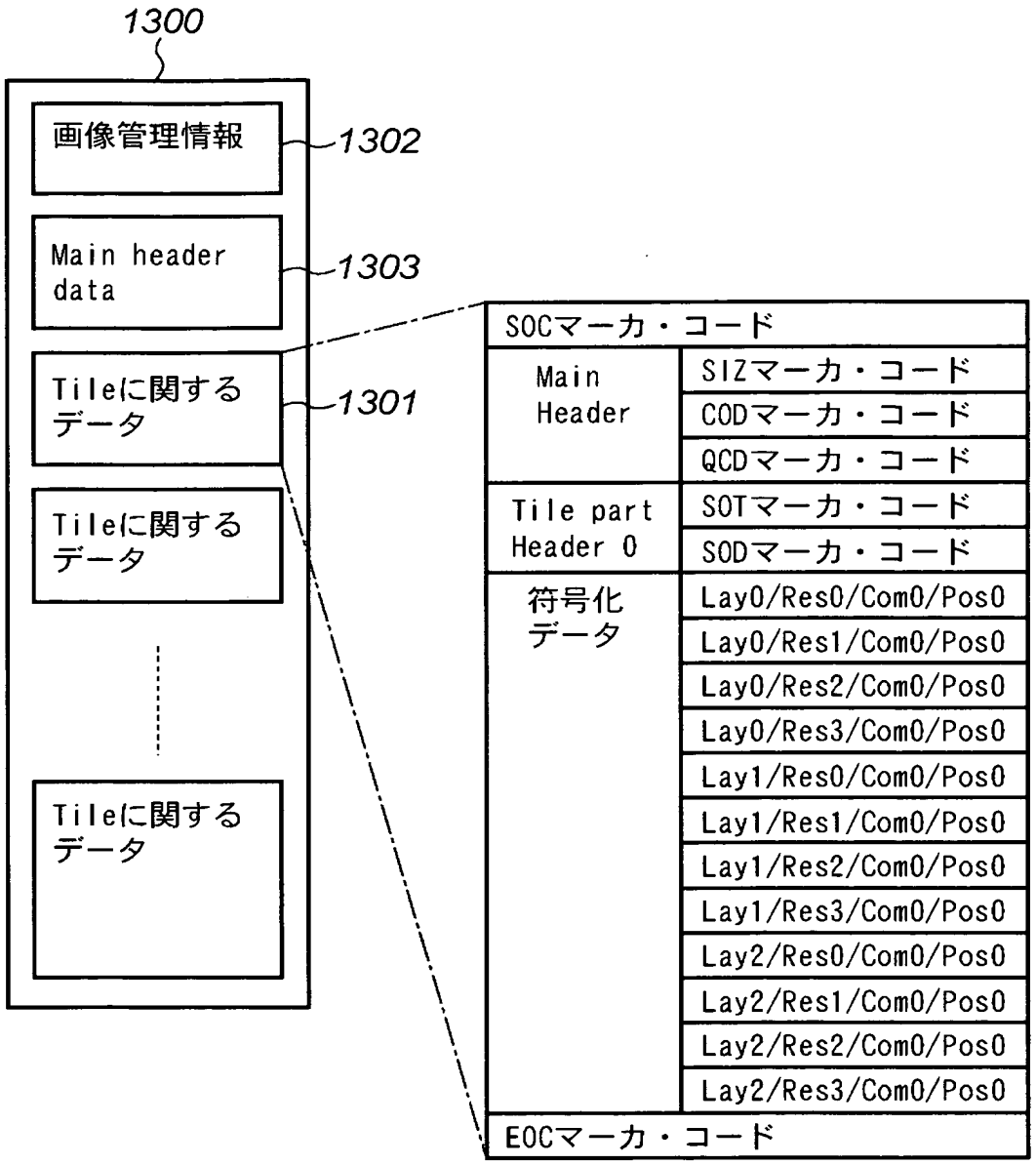
【図１３】


1300
画像管理情報
1302
Main header data
1303
Tileに関するデータ
1301
Tileに関するデータ
Tileに関するデータ
1304
Tileデータの管理用データ
1304a
Tile header
1304b
Packetデータの管理用データ
1304c
Packetデータの管理用データ
Packetデータの管理用データ
1305
Tileのデータ長
1305a
Tile番号
1305b
Packet数
1305c
Resolution数
1305d
Layer数
1305e
Component数
1305f
Position数
1305g
1306
データ長
1306a
ポインタ値
1306b
Packetの番号
1306c
Packetの基本情報
1306d

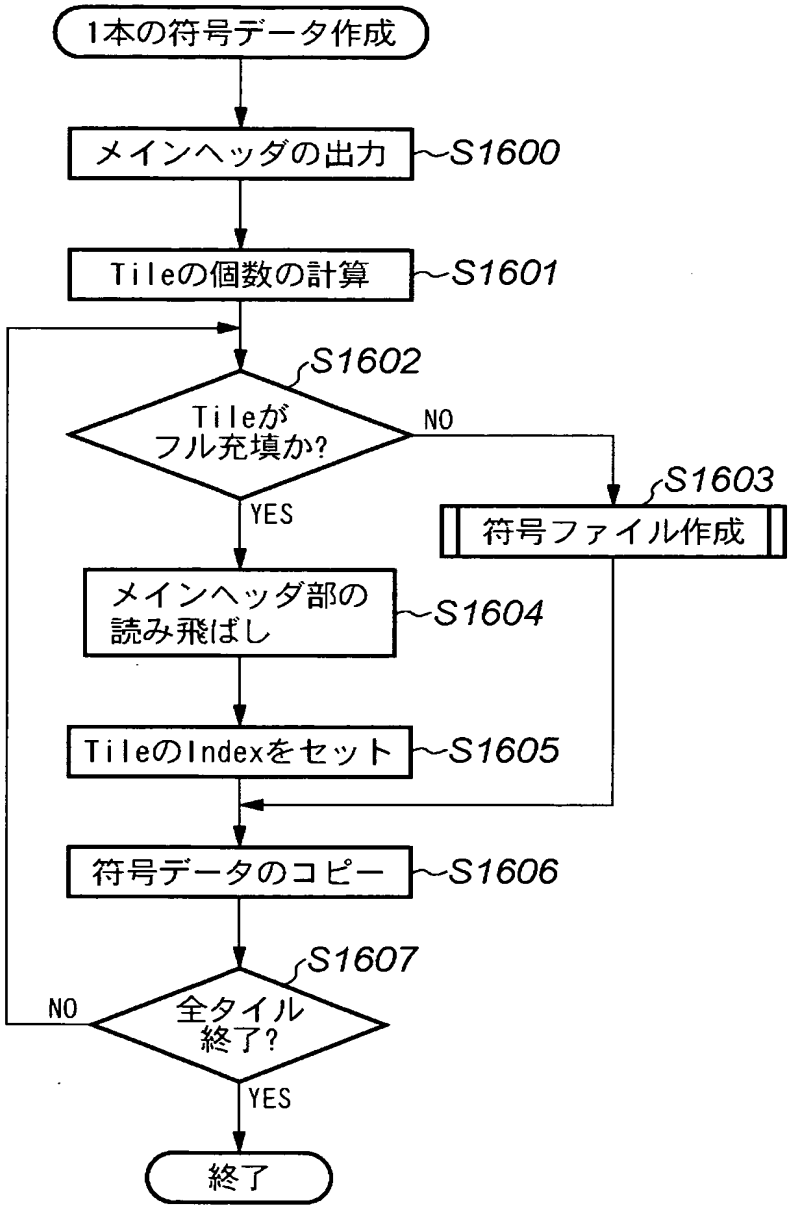
【図１４】


完全符号データ
作成処理
Tileデータ
管理テーブルの削除
S1400
完全符号データを
セット
S1401
終了

【図１５】


1300
画像管理情報
1302
Main header data
1303
Tileに関するデータ
1301
Tileに関するデータ
Tileに関するデータ
SOCマーカ・コード
Main Header
SIZマーカ・コード
CODマーカ・コード
QCDマーカ・コード
Tile part Header 0
SOTマーカ・コード
SODマーカ・コード
符号化データ
Lay0/Res0/Com0/Pos0
Lay0/Res1/Com0/Pos0
Lay0/Res2/Com0/Pos0
Lay0/Res3/Com0/Pos0
Lay1/Res0/Com0/Pos0
Lay1/Res1/Com0/Pos0
Lay1/Res2/Com0/Pos0
Lay1/Res3/Com0/Pos0
Lay2/Res0/Com0/Pos0
Lay2/Res1/Com0/Pos0
Lay2/Res2/Com0/Pos0
Lay2/Res3/Com0/Pos0
EOCマーカ・コード

【図１６】


1本の符号データ作成
メインヘッダの出力
S1600
Tileの個数の計算
S1601
S1602
Tileが
フル充填か?
NO
YES
S1603
符号ファイル作成
メインヘッダ部の
読み飛ばし
S1604
TileのIndexをセット
S1605
符号データのコピー
S1606
S1607
全タイル
終了?
NO
YES
終了

【書類名】　要約書

【要約】

【課題】クライアントにキャッシュされている断片化された符号データと、サーバから必要な分だけ受信した断片化された符号データとを用いて、クライアントにおいて汎用ＪＰＥＧ２０００デコーダで使用可能な符号データを好適に作成することができる符号データ作成方法及び装置を提供する。

【解決手段】　クライアントは、サーバが管理する符号データのうち、第１の符号データを格納しており、ＪＰＥＧ２０００符号データの作成に必要となる符号データと第１の符号データとから、不足する第２の符号データを算出する。そして、サーバから第２の符号データを取得し、そのヘッダ情報を解析して、符号データを複数の独立した符号データに分割する。さらに、分割された単位毎に、独立した符号データの全てのデータが格納されていない場合、ダミー符号データを格納し、その符号データをＪＰＥＧ２０００符号データとする。

【選択図】　図５

認定・付加情報

特許出願の番号	特願２００３－２０１１６２
受付番号	５０３０１２２３６４７
書類名	特許願
担当官	小野寺　光子　　１７２１
作成日	平成１５年　７月２９日

＜認定情報・付加情報＞

【特許出願人】
- 【識別番号】 000001007
- 【住所又は居所】 東京都大田区下丸子３丁目３０番２号
- 【氏名又は名称】 キヤノン株式会社

【代理人】 申請人
- 【識別番号】 100076428
- 【住所又は居所】 東京都千代田区紀尾井町３番６号　秀和紀尾井町パークビル７Ｆ　大塚国際特許事務所
- 【氏名又は名称】 大塚　康徳

【選任した代理人】
- 【識別番号】 100112508
- 【住所又は居所】 東京都千代田区紀尾井町３番６号　秀和紀尾井町パークビル７Ｆ　大塚国際特許事務所
- 【氏名又は名称】 高柳　司郎

【選任した代理人】
- 【識別番号】 100115071
- 【住所又は居所】 東京都千代田区紀尾井町３番６号　秀和紀尾井町パークビル７Ｆ　大塚国際特許事務所
- 【氏名又は名称】 大塚　康弘

【選任した代理人】
- 【識別番号】 100116894
- 【住所又は居所】 東京都千代田区紀尾井町３番６号　秀和紀尾井町パークビル７Ｆ　大塚国際特許事務所
- 【氏名又は名称】 木村　秀二

特願２００３－２０１１６２

出　願　人　履　歴　情　報

識別番号　　［０００００１００７］

１．変更年月日　　１９９０年　８月３０日
［変更理由］　　新規登録
住　所　　東京都大田区下丸子３丁目３０番２号
氏　名　　キヤノン株式会社